Titre: Title:	Utilisation de l'envoi multichemins disjoints pour la sécurité de réseaux ad-hoc mobiles
Auteur: Author:	Anis Mansour
Date:	2006
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Mansour, A. (2006). Utilisation de l'envoi multichemins disjoints pour la sécurité de réseaux ad-hoc mobiles [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7849/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/7849/
Directeurs de recherche: Advisors:	
Programme: Program:	Non spécifié

UNIVERSITÉ DE MONTRÉAL

UTILISATION DE L'ENVOI MULTICHEMINS DISJOINTS POUR LA SÉCURITÉ DE RÉSEAUX AD-HOC MOBILES

MANSOUR ANIS
DÉPARTEMENT DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION DU DIPLÔME DE
MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
DÉCEMBRE 2006

Your file Votre référence
ISBN: 978-0-494-25558-2
Our file Notre référence
ISBN: 978-0-494-25558-2

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

UTILISATION DE L'ENVOI MULTICHEMINS DISJOINTS POUR LA SÉCURITÉ DE RÉSEAUX AD-HOC MOBILES

présenté par : Mansour Anis
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury constitué de :

M. PESANT Gilles , Ph.D., président.
M. FERNANDEZ José M., Ph.D., membre et directeur de recherche.
M. PIERRE Samuel, Ph.D., membre.

Remerciements

Je tiens à remercier mes parents ainsi que mes sœurs pour m'avoir encouragé durant toute la durée de mes études et pour m'avoir apporté un soutien indispensable à ma réussite.

Je tiens avant tout à remercier mon directeur Monsieur José M. Fernandez pour son soutien logistique, financier ainsi que pour ses bons conseils sans lesquels il m'aurait était impossible de mener à terme ce projet de recherche.

Je tiens aussi à remercier Faiza KACEM qui n'a cessé de me supporter tout au long de ce travail. Je remercie aussi mon ami Chiheb DEKHIL qui en l'espace d'un après midi m'a fait gagner plus d'un mois de travail.

Résumé

La mobilité touche aujourd'hui une multitude d'applications et les réseaux informatiques ne font pas d'exception. En effet, l'informatique qui est de plus en plus présente et indispensable dans nos vies, trouve ses limites quand il s'agit de se mouvoir. L'absence d'une infrastructure étendue comme dans le cas de la téléphonie mobile, a fait voir le jour à des réseaux informatiques mobiles sans aucune infrastructure préétablie, ce sont les réseaux Ad-hoc mobiles ou RAHM. Au début, le besoin de ce type de réseau était lié à des applications de type militaire ou encore à des situations d'urgences telles que les catastrophes naturelles, pour être étendu à des applications plus conventionnelles. Un RAHM est basé sur un ensemble d'entités mobiles auto organisées et sans infrastructure prédéfinie, communicant ensemble en relayant l'information entre les nœuds distants. Les nœuds assument donc la tâche de routage des paquets ainsi que l'acheminement des paquets en provenance d'autres nœuds.

Cette collaboration permet la communication entre nœuds au-delà de la portée limitée de leurs antennes, ce qui donne une certaine flexibilité et ne limite pas la taille du réseau. Malheureusement la nature de ce concept, très différent des réseaux conventionnels, fait sa force mais aussi sa faiblesse. En effet, cette flexibilité engendre de sérieux problèmes liés à la sécurité, exposant ce concept à une multitude de vulnérabilités.

Dans le cadre de cette recherche, nous avons essayé de traiter les problèmes entourant la sécurité des réseaux Ad-hoc mobiles et, plus précisément, ceux reliés à la garantie de l'intégrité des données circulant sur ce type de réseau. Nous avons pensé à utiliser la technique d'envoi multichemins disjoints pour l'acheminement des données et la garantie de leur intégrité. À cet effet, nous nous sommes basés sur le protocole de routage multichemins disjoints AOMDV pour proposer un protocole sécurisé visant à améliorer cet aspect. Le protocole AOMDV tente d'exploiter et de mettre à profit une des propriétés particulières des réseaux Ad-hoc mobiles qui n'est autre que l'existence de plusieurs chemins reliant un nœud source à un nœud destination. Il s'agit de chercher et de maintenir un ensemble de routes disjointes. Le protocole SDMRP que nous proposons utilise un sous-ensemble de taille n de ces routes dis-

jointes afin de créer une redondance permettant de vérifier l'intégrité des données. L'information de redondance consiste en un haché de l'information à envoyer, qui a la particularité d'avoir une taille très réduite permettant ainsi d'éviter l'encombrement du réseau. L'idée des routes disjointes évite la présence d'un même nœud malicieux sur deux routes différentes. Cette propriété force un éventuel attaquant à un plus grand déploiement sur le réseau et l'oblige à assurer la collaboration des nœuds qui sont en sa possession pour pouvoir placer une attaque, ce qui rend cette tâche très ardue.

Pour pouvoir évaluer les performances du protocole proposé, nous avons, en premier lieu, modélisé mathématiquement son comportement par rapport à la garantie de l'intégrité des données et aussi du point de vue de la qualité de service. Nous avons, par la suite, implémenté et simulé ce protocole et avons mis l'accent sur les mêmes deux aspects, à savoir la probabilité de succès d'une attaque sur l'intégrité et la latence moyenne bout à bout. Le protocole de référence utilisé est AODV qui est à l'origine du protocole AOMDV sur lequel est basé le protocole SDMRP proposé.

Les résultats obtenus démontrent une très grande amélioration concernant la garantie de l'intégrité des données puisque la probabilité de succès d'une attaque décroit exponentiellement avec le nombre n de routes alors que le délai moyen reste raisonnable pour une transmission de données et n'augmente que légèrement avec n.

Abstract

Mobility has become a main feature for several applications and computer networks are not an exception. In fact, computing which is more and more present and essential find its limits when it has to move.

The lack of a wide spread infrastructure such as the one we can find in cellular networks, leaded to set up mobile networks without any fixed infrastructure commonly named Ad-hoc mobile networks or MANETs. Originally, those networks were designed for military applications and emergency situations such as natural disasters where an important need to share information without any predefined infrastructure is present to be extended to more conventional applications such as meetings, forums, etc.

Mobile Ad-hoc networks consist of a collection of wireless mobile nodes self-organised and without any predefined infrastructure, exchanging data among themselves without the reliance on a fixed base station or a wired backbone network. Nodes also assume the packets routing task. This particularity allows nodes to communicate further than their limited antenna range and gain flexibility without limiting the network size.

This concept, very different from the conventional networks, presents a real improvement to the wireless networking field. Unfortunately, it also presents several weaknesses mostly related to data security.

In this thesis, we tried to solve the data security problem and specially data integrity in the mobile ad-hoc networks. We propose a strategy consisting in developing a secure protocol based on the multipath disjoined nodes routing to preserve the data integrity from modification.

Our approach consists in using a particular MANET property traduced by the availability of a multiple different and node disjoint routes from a source node to a destination one. The proposed protocol finds and maintains a set of routes between a source and a destination node. It then uses a subset of those routes to send data and creates a redundancy allowing the integrity check. The redundancy data consist in the hash of each sent packet. Due to their little sizes, the hashes prevent the network congestion.

The disjoint nodes routing protocol provide a security against the presence of the same malicious node on two or more different routes. This strategy forces the attacker to deploy more than one network node and to insure their collaboration for any eventual successful attack. This task is not as easy as it sounds to be and needs a colossal deployment effort to guaranty any attack feasibility.

To evaluate the performances of the proposed protocol SDMRP (Secure Disjoint Multipath Routing Protocol), we implemented it and simulated a series of experiments and scenarios. We focused our experimental effort on evaluating the probability of a successful attack and the mean end to end latency. We compared our protocol to AODV (Ad-hoc On-demand Distance Vector routing) which was used as a basis to build SDMRP.

We have obtained interesting results, in particular for the probability of attack success which decreases as we increase the number of used routes. The mean end to end latency was also reduced compared with AODV.

Table des matières

Liste des figures

Liste des sigles

Abréviation	Signification
RAHM	Réseau Ad-Hoc Mobiles
MANET	Mbile Ad-hoc Network
DSR	Dynamic Source Routing
AODV	Ad-hoc On-demand Distance Vector routing
TBRPF	Topology Broadcast Based on Reverse-Path Forwarding
DSDV	Destination Sequenced Distance-Vector
ZRP	Zone Routing Protocol
BRP	Broadcast Resolution Protocol
SDMP	Secured Data based Multipath Protocol
CONFIDANT	Cooperation Of Nodes - Fairness In Dynamic Ad-hoc NeTworks
CORE	COllaborative REputation
WPA	Wi-Fi Protected Access
WEP	Wired Equivalent Privacy
AOMDV	Ad-hoc On-demand Multipath Distance Vector routing
SDMRP	Secure Disjoint Multipath Routing Protocol

CHAPITRE 1

Introduction

Depuis maintenant quelques années, les réseaux Ad-hoc mobiles (RAHM) ou encore MANET (mobile Ad-hoc Network) commencent à susciter une attention particulière, non pas de la part de la communauté scientifique seulement mais aussi de la part de la communauté industrielle. Cet intérêt peut être interprété comme un signe évident d'une prochaine émergence pour ce type de réseaux.
Les réseaux Ad-hoc mobiles se basent sur l'idée d'un réseau mobile sans infrastructure préétablie, cette particularité représente le point fort de ce type de réseau. La simplicité ainsi que la rapidité de leurs déploiements à des endroits où il est impossible de mettre en place une infrastructure permanente ont fait d'eux les meilleurs candidats pour répondre à des situations bien particulières tel que le déploiement sur un site où il vient de se produire un catastrophe naturelle ou sur le champ d'un déploiement militaire. Dans les deux cas, le besoin de communiquer est évident et seuls les RAHM peuvent répondre à ce besoin.
Dans les années passées, tous les efforts de recherche destinés aux réseaux Ad-hoc mobiles étaient pratiquement dirigés vers la mise en place de protocoles de routages adaptés à la situation particulière des RAHM, soit la mobilité, les protocoles de routages classiques ne pouvant répondre aux exigences d'un réseau dynamique. Il est évident qu'un routage adéquat et adapté revêt une très grande importance pour permettre une exploitation correcte de ce type de réseaux. Toutefois, il existe d'autres points d'intérêts qui sont tout autant important que le routage dynamique, en particulier tous les points portant sur la sécurité dans les RAHM. En effet, sans une garantie de sécurité, aucun type de réseau ne peut se développer et en particulier les RAHM. À cause de leur nature, ces réseaux posent des problèmes de sécurité jusqu'à la inexistants dans les autres types de réseaux dits conventionnels.
Le fait que les RAHM se basent sur le concept d'un ensemble d'entités mobiles auto organisées et sans infrastructure prédéfinie, communicants ensembles en relayant l'information entre les nœuds distants, où chacun des nœuds du réseau joue le rôle d'un

routeur et où n'importe quel nouveau nœud peut faire partie du réseau sans qu'il n'y est aucune autorité centralisée, est à l'origine d'innombrables vulnérabilités. Il est donc très important de se pencher sur ces problèmes de sécurité et en particulier la garantie de l'intégrité des données circulant sur le réseau. Ceci constitue l'objet de ce mémoire.

1.1 Concepts de base en réseau Ad-hoc

Contrairement à tous les autres types de réseaux, et même ceux qui sont considérés mobiles tel que les réseaux de téléphonie cellulaire, les réseaux Ad-hoc mobiles ne se basent pas sur une infrastructure dorsale dite *backbone*. Il est à noter que dans les réseaux de type téléphonie cellulaire, seuls les terminaux sont mobiles, le réseau lui reste fixe, ce qui n'est pas le cas avec les RAHM. Ce sont les nœuds qui représentent le réseaux, alors si les nœuds sont mobiles le réseau est lui-même mobile. Ceci constitue une différence majeure avec tout autre type de réseau fixe ou mobile.
Dans les réseaux Ad-hoc mobiles, la définition du routage est quelque peu différente de la définition conventionnelle employée dans les autres types de réseaux. Une route est définie comme un ensemble d'entités mobiles impliquées dans la bonne transmission des données entre un nœud source et un nœud destination. Le routage n'est autre qu'une opération de sélection et de découverte de routes entre les paires source/destination. Les nœuds étant mobiles, la duré de vie d'une route est donc très limitée. Les protocoles de routage doivent tenir compte de cette particularité pour garantir une bonne qualité de service. Le maintien des routes revêt une très grande importance, c'est ce qui différencie un bon algorithme de routage d'un mauvais.
Il est important de signaler l'absence d'une autorité centralisée. D'ailleurs, c'est contraire à l'idée de base des réseaux Ad-hoc mobiles. Un pareil type de nœud serait dans un emplacement permanant et fixe, où tous les autres nœuds pourraient le joindre ; c'est le rôle que jouent les stations de base (*base station*) dans les réseaux de téléphonie cellulaire.
L'absence d'une autorité centralisée crée une situation difficile à gérer. N'importe quel nœud peut donc faire partie du réseau et profiter ainsi de ses ressources, c'est d'ailleurs le but requis. Cette ouverture qui fait la force des RAHM, est à l'origine de la majorité des problèmes de sécurité entourant ce type de réseau tel que, l'absence d'authentification,de la confidentialité, de l'intégrité ainsi que de la non répudiation.

1.2 Éléments de la problématique de sécurité

Face aux ambitions que suscitent les réseaux Ad-hoc mobiles, ainsi qu'à la récente multiplication des applications utilisant ce type de réseau, il est impératif de répondre et de satisfaire aux exigences que requièrent ces nouvelles applications.
Les défis à relever sont multiples et touchent différents domaines de recherche. La garantie de la survie ainsi que du succès des réseaux Ad-hoc mobiles passe donc par leur capacité à s'adapter et à répondre efficacement aux nouvelles exigences tel que la garantie d'une certaine qualité de service, ou encore, la garantie de la sécurité.
Répondre aux exigences concernant la sécurité dans les RAHM est devenu un enjeu majeur et incontournable autant pour la communauté scientifique que pour la communauté industrielle. Cependant, ce problème de sécurité revêt plusieurs facettes, rendant sa prise en charge très ardue. En effet, les problèmes entourant la sécurité sont assez hétérogènes et ne sont donc pas tous aisément satisfaits. De par leur nature, les réseaux Ad-hoc mobiles imposent de multiples contraintes de sécurité.
Le médium radio est utilisé pour supporter les communications, or ce type de médium est connu pour être peu fiable. Il ne procure d'ailleurs aucune protection exposant ainsi toutes les communications à l'écoute. La topologie entièrement dynamique, basée sur les nœuds du réseaux, empêche la présence d'une quelconque autorité de gestion ou de certification centralisée.
Cette absence rend très difficile, voir impossible, la distribution de certificats d'authentification ou des clés pour l'encryption des transmissions. Il est à noter que n'importe quel nœud se trouvant à porté des antennes des autres nœuds constituant le réseau, fera automatiquement partie de ce dernier. La puissance limitée de calcul dont disposent les nœuds rend difficile la mise en place de protocoles cryptographiques assez sécuritaires. Les RAHM souffrent aussi d'autres types problèmes tel que la quantité limitée d'énergie dont disposent les nœuds. En effet, étant mobiles, les nœuds ne disposent pas de sources d'énergie autre que leur batteries, ce qui les pousse à un comportement égoïste. Ce comportement est conforté par la faiblesse de la bande passante dont ils disposent.
Ces différences et ces limitations font en sorte qu'il est difficile voir impossible d'utiliser ou même d'adapter les solutions de sécurité déployées sur les réseaux conventionnels. Il existe plusieurs travaux proposant différentes solutions aux différents problèmes reliés à la sécurité des réseaux Ad-hoc mobiles.

Ces solutions tentent de résoudre les problèmes liés à la disponibilité des services, à la confidentialité des données, à l'authentification des nœuds et à la non répudiation des données transmises. Toutefois, la garantie de l'intégrité des données transmises par des nœuds intermédiaires n'a pas suscité autant d'intérêt qu'elle le devrait. En effet, le problème de la garantie de l'intégrité des données trouve tout son sens dans les réseaux Ad-Hoc mobiles car dans ce type de réseau, les nœuds intermédiaires jouent le rôle de routeurs et relaient donc l'information entre les nœuds source et destination. Chacun de ces nœuds intermédiaires pourrait, à tout moment, décider d'apporter des modifications aux données qu'il relaye sans que les nœuds source et destination ne s'en rendent compte. Cette composante de la sécurité des réseaux est donc indispensable pour les réseaux Ad-Hoc.

1.3 Objectifs de recherche

Le principal objectif de cette recherche est d'étudier et de développer un protocole de routage sécurisé permettant d'améliorer la garantie de l'intégrité des données transitant sur les réseaux Ad-hoc mobiles.
À cet effet nous voulons atteindre les objectifs suivants :

- Analyser et évaluer les solutions existantes en matière de sécurité dans les réseaux Ad-hoc. Plus particulièrement en ce qui concerne la garantie de l'intégrité des données.
- Concevoir et implémenter un protocole de routage permettant d'améliorer la garantie de l'intégrité des données tout en essayant de ne pas affecter la qualité de service. Ce protocole se base sur l'utilisation d'envoi redondant des données sur de multiples routes disjointes. Les informations sont transmises sur chacune des routes de façon à forcer l'attaquant à multiplier son déploiement sur le réseau et de s'assurer la collaboration des nœuds sous son contrôle. À cet effet, nous prévoyons l'utilisation et l'adaptation d'un protocole de routage multichemins disjoints existant.
- Évaluer les performances du protocole proposé par rapport aux protocoles de référence cités dans la littérature par le biais de l'analyse mathématique et de la simulation.
- Étudier les compromis entre l'amélioration de la garantie de l'intégrité des données et la dégradation de la qualité de service (QoS) qui pourrait en découler

de notre solution proposée.

1.4 Plan du mémoire

Après avoir développé le chapitre en cours nous allons nous concentrer sur quatre autres chapitres.

- Le chapitre deux sera consacré à la revue de littérature. Il fera l'état des travaux traitant de la sécurité dans les réseaux Ad-hoc mobile et plus précisément ceux consacrés à la garantie de l'intégrité des données. Il traitera aussi des solution de routage multichemins disjoints.
- Le chapitre trois sera, quant à lui, consacré à l'introduction de la solution proposée qui aura la tâche d'améliorer la garantie de l'intégrité des données par l'utilisation de l'envoi multichemins disjoints. Ce chapitre contiendra une description détaillée des routines et des algorithmes composant ce protocole ainsi que sa modélisation mathématique.
- Dans le chapitre quatre nous allons présenter les détails d'implémentation du protocole proposé ainsi que les détails des simulations à faire. Ces simulations nous permettront de confronter notre protocole à diverses situations permettant ainsi d'évaluer ses performances et de les comparer aux performances des protocoles existants.
- Enfin le chapitre cinq sera consacré aux conclusions ce qui permettra de faire la synthèse du travail réalisé. De cette synthèse se dégageront les principales contributions apportées par ce travail de recherche et aussi la mise au point sur les limites des ces travaux. Finalement, les directions de recherche pour les travaux futurs.

CHAPITRE 2

Revue de littérature

La mobilité touche aujourd'hui une multitude d'applications, les réseaux informatiques ne font pas d'exception. En effet, l'informatique qui est de plus en plus présente et indispensable dans nos vies, trouve ses limites quand il s'agit de se mouvoir.

L'absence d'une infrastructure étendue comme dans le cas de la téléphonie mobile, a fait voir le jour à des réseaux informatiques mobiles sans aucune infrastructure préétablie, ce sont les réseaux Ad-hoc mobiles (RAHM). Au début, le besoin de ce type de réseau était lié à des applications de type militaire ou encore à des situations d'urgences tel que les catastrophes naturelles, pour être étendu à des applications plus conventionnelles.

Un RAHM est basé sur un ensemble d'entités mobiles, auto organisées et sans infrastructure prédéfinie, communicant ensembles en relayant l'information entre les nœuds distants. Les nœuds assument donc la tâche de routage des paquets ainsi que l'acheminement des paquets en provenance d'autres nœuds. Cette collaboration permet la communication entres nœuds au-delà de la porté limitée de leurs antennes, ce qui donne une certaine flexibilité et ne limite pas la taille du réseau.

Malheureusement, la nature de ce concept, très différent des réseaux conventionnels, fait sa force mais aussi sa faiblesse. En effet, cette flexibilité engendre de sérieux problèmes de sécurité et ouvre la porte à une multitude de vulnérabilités.

Dans ce chapitre, nous allons donner un aperçu des différents aspects liés à la sécurité et à la vulnérabilité des RAHM, ainsi que les différents mécanismes et contre-mesures déployés afin de rendre les RAHM plus robustes et plus sécuritaires.

2.1 Réseaux Ad-hoc mobiles - RAHM

Contrairement aux réseaux conventionnels, les RAHM ne disposent d'aucune infrastructure dédiée. Les nœuds sont amenés à jouer des rôles importants, tel que le routage et la découverte de services. D'autre part, le medium de transmission utilisé

n'est autre que l'air et les nœuds sont en perpétuel mouvement. La topologie du réseau change donc constamment.

2.1.1 Routage dans les RAHM

Tout comme pour les réseaux conventionnels, les protocoles de routage dans les réseaux Ad-hoc mobile se divisent en deux familles : Les protocoles de routage réactifs et les protocoles de routage proactifs.

2.1.1.1 Protocoles de routage réactifs

Un protocole de routage réactif fonctionne selon le principe suivant : Si un nœud donné a le besoin de communiquer avec un nœud qui se trouve au-delà de sa portée, il doit lancer une recherche de route pour le joindre. Ce qui implique que les routes sont construites à la demande et qu'il n'y a pas de demandes périodiques de route. Ceci engendre une certaine latence due à la recherche de la route. Les protocoles de routage réactifs les plus connus sont *Dynamic Source Routing* (DSR) [Johnson et Maltz (1996)] et *Ad-hoc On-demand Distance Vector routing* (AODV) [Perkins (1997)].

- **Protocole DSR :** C'est au niveau du nœud émetteur que se fait le choix de routage d'un paquet donné. La route que le paquet suivra est spécifiée dans l'en-tête du paquet lors de son émission. La détection des routes entre deux nœud A et B se fait à l'aide du mécanisme *Route Discovery Protocol.* Si le nœud A(nœud émetteur) ne connaît pas la route lui permettant d'atteindre le nœud B(nœud destination), un paquet *Route Request* est diffusé dans le réseau. Il contient un *Request ID* spécifique à la requête et à l'émetteur. Il est alors retransmit par les autres nœuds s'il est reçu pour la première fois jusqu'a atteindre le nœud destination. À ce moment, un paquet *Route Reply* contenant la liste des nœuds empruntés est retourné au nœud émetteur A. Si, à cause des changements dus à la topologie dynamique du réseau, la route devient incorrecte, un mécanisme nommé *Route Maintenance Protocol* prend la relève en émettant un paquet *Route Error.*
- **Protocole AODV :** Ce protocole est tiré de DSDV mais il diffère par le fait d'être complètement réactif. La détection des routes se fait par l'émission de paquets *Route Request* tout comme pour DSR, à la différence près que le

mécanisme ne stocke pas la route dans l'entête des paquets *Route Request*, mais la distribue au niveau de chaque nœud . Afin d'éviter les diffusions multiples pour une même requête, AODV utilise un identifiant de diffusion *Broadcast ID*. Il est agrémenté d'un numéro de séquence source et d'un numéro de séquence destination. Ces deux numéros de séquences permettent respectivement de connaître l'état de péremption des donnés de routage à la source, et de déterminer les données de routage que la source peut accepter. AODV fait en sorte que si un nœud a en mémoire un numéro de séquence destination plus petit que celui reçu, il le transmet à son voisinage sans y répondre. Un chemin inverse (Reverse Path) est donc construit permettant ainsi le routage des paquets de réponse vers la source. Ces paquets jouent le rôle d'un pointeur qui associe le nœud qui a émis la requête de route au numéro de séquence de la source.

2.1.1.2 Protocoles de routage proactifs

Contrairement aux protocoles de routage réactifs, les protocoles de routage proactifs maintiennent en permanence et périodiquement les informations concernant la topologie du réseau. Ceci est à l'origine d'une signalisation assez encombrante et assez lourde pour le réseau. Les protocoles de routage proactifs les plus connus sont *Optimized Link State Routing protocol* (OLSR) [Clausen *et al.* (2003)], *Topology Broadcast Based on Reverse-Path Forwarding* (TBRPF) et *Destination Sequenced Distance-Vectro* (DSDV) [Perkins et Bhagwat (1994)].

- **Protocole OLSR :** Le protocole OLSR fait partie de la famille des protocoles à état de liens. Une différence le distingue cependant de sa famille d'origine, car il utilise une topologie partielle du réseau. En d'autres termes, les nœuds ne diffusent pas tous les liens qu'ils ont avec leurs voisins, mais diffusent seulement un sous-ensemble de ces derniers, ce qui permet de les joindre. La taille des paquets de contrôle est donc significativement réduite. OLSR est basé sur la technique des relais multipoint qui permet d'optimiser la diffusion des messages de contrôle et ainsi économiser de la bande passante. Le routage se fait par sauts, en se basant sur les informations collectées sur les paquets de contrôle reçus, et chaque nœud calcule ainsi sa table de routage.
- **Protocole DSDV :** Le Protocole DSDV est parmi les premiers protocoles spécialement dédiés aux réseaux ad-hoc. C'est un protocole vecteurs de distance

(*distance-vectors*). Il utilise des numéros de séquences (*sequence number*) afin de découvrir la plus récente route. Un nombre est associé à chaque route vers un nœud appartenant au réseau. Périodiquement, chaque nœud inonde ses voisins avec le contenu de sa table de routage. Il transmet, en outre, des mises à jour s'il y a une modification de la topologie de son voisinage.

2.1.1.3 Protocoles de routage Hybrides

Les protocoles de routage hybrides sont des protocoles qui combinent les deux types de protocoles précédents, les proactifs et les réactifs. Cette combinaison essaye de profiter des avantages de ces deux types tout en évitant leurs inconvénients. Parmi ces protocoles le *Zone Routing Protocol*(ZRP) [Haas *et al.* (2002)].

- **Protocole ZRP :** Le protocole ZRP est basé sur deux protocoles, à savoir le protocole de routage intrazone (IARP) et le protocole de routage interzone (IERP). Le réseau est donc découpé en zones qui peuvent s'inter couper. La communication dans une zone donnée se fait grâce au routage intrazone (IARP), tandis que la communication entre zones se fait grâce au routage interzone (IERP). La liaison entre (IARP) et (IERP) elle se fait, quant à elle, par un troisième composant nommé *Broadcast Resolution Protocol*(BRP). Si un nœud A veut atteindre un nœud B, deux cas se présentent, le nœud A est dans la même zone que le nœud B, dans ce cas le IARP est utilisé, sinon c'est IERP qui est utilisé.

2.1.1.4 Protocoles de routage Multi-chemins

Ces types de protocoles sont très peu utilisés. Ils se basent sur un routage utilisant plusieurs chemins reliant une source S et une destination D [Li et Cuthbert (2004), Lee et Gerla (2001), Das et Marina (2001)]. L'idée derrière ce choix est de calculer plusieurs routes entre chaque paire de nœuds et de garder ensuite une liste des routes reliant ces paires de nœuds. L'algorithme n'utilise qu'une route parmi les routes de la liste à la fois. Cette liste trouve son utilité quand la route en cours d'utilisation expire, c'est-à-dire que la route n'est plus valable. En effet, contrairement aux algorithmes de routage conventionnels, l'algorithme de routage multichemis ne lance pas une nouvelle découverte de route chaque fois qu'une route est brisée mais puise dans la liste de routes déjà calculées pour remplacer la route expirée en espérant qu'il existe

encore des routes valables dans la liste.
Contrairement aux routes dans les réseaux conventionnels, les routes sur les réseaux Ad-hoc mobiles ont des durées de vie très limitées. La mobilité des nœuds dans les RAHM se traduit par une topologie dynamique. L'avantage de garder plusieurs routes dans une liste permet d'éviter au maximum les délais dus à la perte des routes et de réduire la signalisation très lourde reliée à la découverte de routes. Dans la majorité des cas, une route brisée est tout de suite remplacée sans délai ni signalisation.

2.1.2 Contraintes des RAHM

De part leur différence fondamentale avec les réseaux conventionnels, les réseaux Ad-hoc mobiles sont accompagnés de plusieurs contraintes. Dans [Corson et Macker (1999)], ces contraintes caractéristiques aux RAHM sont classées en différents types. Ce sont les contraintes de bande passante, les contraintes de topologies dynamiques, les contraintes de la sécurité physique, les contraintes d'énergie et les contraintes liées au vaste déploiement du réseau.

- **Contraintes de bande passante :** Ayant une capacité en bande passante inférieure à celle des liens câblés, les liens sans fils cumulent d'autres handicaps liés aux interférences, au bruit, aux accès multiples, à l'atténuation du signal, etc.. Ce qui diminue la qualité de service et facilite les attaques de dénis de service.
- **Contraintes de topologie dynamique :** Comme cité auparavent, les nœuds sont mobiles et peuvent se déplacer à tout moment et dans n'importe quelle direction. N'ayant pas les mêmes capacités d'antennes, les liens entre nœuds peuvent passer de bidirectionnels à unidirectionnels. Le réseau change donc constamment de topologie, ce qui nécessite des protocoles de routage spécifiques aux RAHM, tel que détaillés précédemment.
  Contraintes de la sécurité physique : Basés sur les communications sans fils, les RAHM sont vulnérables à l'écoute, puisqu'il suffit de mettre une antenne pour capter les transmissions. Ils sont aussi vulnérables au brouillage qui généralement entraîne un dénis de service.
- **Contraintes d'énergie :** Les nœuds dans un RAHM sont généralement mobiles et déployés dans des endroits où ils n'ont pas accès à de l'énergie.

Ils fonctionnent donc sur leurs batteries et ont alors des autonomies limitées. Ceci engendre un comportement égoïste. En effet, pour conserver leur énergie, les nœuds vont s'abstenir de relayer les paquets qui ne leurs sont pas destinés.

- **Contraintes liées au vaste déploiement du réseau :** Les nœuds dans un RAHM n'ont pas tous les mêmes capacités en terme de mémoire par exemple. Les RAHM n'ont pas une taille préfixée et le réseau peut prendre des proportions énormes. Étant donné que les nœuds ont pour mission de router les paquets, ils sont supposés garder en mémoire une information qui grandit avec la taille du réseau, ce qui peut devenir contraignant pour certains nœuds.

2.2 Sécurité des RAHM

Comme cité auparavent, la sécurité dans les réseaux Ad-hoc mobiles pose un sérieux défi. Ceci est relié à la nature du concept de ces dernier. Mais avant de parler de sécurité, nous allons introduire les notions qui la composent communément appellés les services de la sécurité. Dans [Stallings (2002)], cinq service sont énumérés, la disponibilité, la confidentialité, l'intégrité, l'authentification et la non répudiation.

- **La disponibilité :** C'est le fait de maintenir le service disponible dans toutes les situations.
- **La confidentialité :** C'est le fait de contrôler l'accès aux données circulant sur le réseau.
- **L'intégrité :** C'est le fait de garantir la non modification des données circulant sur le réseau.
- **L'authentification :** C'est le fait de s'assurer de l'identité reliée à une communication sur le réseau.
- **La non répudiation :** C'est le fait qu'un émetteur ne puisse pas renier son identité après l'avoir déclaré.

2.2.1 Classifications des attaques

Les attaques sur la sécurité peuvent êtres classées suivant leur nature et suivant les moyens déployés afin de les mener. Dans la littérature ces attaques sont divisées en quatre catégories : Les attaques passives, les attaques actives, les attaques internes et les attaques externes.

- **Attaques passives :** Dans [Michiardi et Molva (2006)] une attaque est dite passive dans deux cas : Si l'attaquant se contente d'écouter sur le réseau ceci est facilement réalisable a l'aide d'une antenne omnidirectionnelles. Sinon, si l'attaquant se contente de ne pas relier l'information et ne pas l'acheminer à destination. Ce comportement est décrit dans certaines références comme plutôt égoïste que passif.
- **Attaques actives :** Contrairement aux attaques passives dans [Michiardi et Molva (2006)], une attaque est dite active s'il y a émission de données de la part de l'attaquant, tel que l'introduction de paquets corrompus ou l'émission continue de paquets en vu de limiter l'accès à une ressource.
- **Attaques internes :** Dans ce cas de figure, l'attaquant doit faire partie du réseau à attaquer, ou doit prendre le contrôle d'un nœud légitime de ce dernier. L'attaquant a donc plus de possibilités et d'influence sur le réseau.
- **Attaques externes :** L'attaque de type externe a moins d'impact sur le réseau. L'attaquant doit se contenter d'attaques de type passif, comme par exemple l'écoute, le brouillage, etc.

Dans la littérature nous avons rencontré certaines références [Golle *et al.* (2004)] qui classifient les attaques suivant d'autres critères comme la **nature** de l'attaque, sa **cible**, sa **portée** ou encore son **impact**. Une attaque peut avoir une cible locale ou distante, ce qui définit la capacité de l'attaquant, car étant distant, l'attaquant va dépendre de la qualité du lien qui le relie au réseau attaqué. D'autre part, les attaques ont différentes portés, elles peuvent êtres limitées ou étendues, suivant le nombres de nœuds compromis. Enfin les attaques ont aussi différents impacts sur le réseau cible. On distingue trois catégories : Les attaques non détectées, les attaques détectées et les attaques réparées.

2.2.2 Les attaques

Dans cette partie nous allons présenter les attaques les plus citées dans la littérature. Comme les attaques ont été classifiées, commençons alors par les attaques dites passives.

Attaques passives : Les attaques passives sont les plus simples, elles sont non détectables et sont une base nécessaire à tout autre types d'attaques.

- **L'écoute :** Le support de communication dans les RAHM n'étant autre que l'air, cette attaque est la plus facile à réaliser. Il suffit à l'attaquant de se trouver à portée d'antenne et d'écouter les communications entres nœuds.
- **Analyse et décryptage de données :** Les communications entre nœuds sont généralement cryptées, l'écoute ne suffit donc pas. Pour cela, l'analyse des données est nécessaire. L'attaque contre la couche de cryptage WEP est un bon exemple. Cette couche présente une faiblesse facilement exploitable. En effet, la clé n'est pas assez longue et elle est constamment réutilisée. Cette attaque est très dangereuse car elle est non détectable et met en échec le système de confidentialité dans le réseau.

Attaques actives : Contrairement aux attaques passives, ce type d'attaques se fait avec émission de paquets ce qui, en théorie, le rend détectable. Mais en pratique ces attaques ne le sont souvent pas.

- **Dénis de service :** Ce type d'attaques demande en général de l'attaquant des ressources considérables. Pour le cas spécifique des RAHM, un brouillage des ondes électromagnétiques suffit à placer une attaque de ce type. Vu que le réseau est sans fil, il est aussi possible de placer une attaque visant les ressources énergétiques limitées des nœuds constituant ce dernier. Cela se fait facilement par la génération d'un trafic assez important sur le réseau dans le but de réveiller assez souvent les nœuds et de les obliger ainsi à consommer leur énergie. Dans [Stajano et Anderson (1999)], cette attaque est appelée torture par la privation de sommeil (*sleep deprivation torture*). Une autres façon serait d'utiliser une autre faiblesse spécifique aux RAHM concernant la nature des nœuds le composant. En effet, ces nœuds sont hétérogènes et n'ont en général pas une grande capacité de calcul. Ceci permet de placer une attaque par la consommation des ressources de calcul. Dans la littérature, cette attaque porte le nom de famine de ressources de calcul.
- **Rejeu ou répétition de paquets :** C'est une attaque assez simple, elle se base sur l'idée de réémettre des paquets déjà reçus. Ceci pose des problèmes de signalisation dans la majorité des algorithmes de routage. Dans [Hu et Perrig (2004)] un exemple de ce type d'attaque est le *Rushing attack*, ou attaque par précipitation. C'est une attaque qui vise les protocoles de routage réactifs par la dissémination de paquets *Route request* créés sur le long du réseau attaqué.

Ceci avant la propagation des vrais *Route request*. Par conséquence, les vrais *Route request* sont rejetés et le protocole est paralysé.

- **Trou de vers :** L'attaque par trou de vers (*wormhole attacks*) [Hu *et al.* (2003a)] est une attaque qui consiste en la création d'un lien tunnel entre deux nœuds attaquants non voisins au moyen d'une liaison câblée ou encore d'un antenne surpuissante. Ceci rend la route utilisant ce tunnel une route privilégiée avec un plus gros débit et moins de sauts. La topologie du réseau change donc en conséquences. Suite à quoi, l'attaquant commence à acheminer les paquets de contrôle seulement et omet volentairement les paquets de données. Si ces paquets étaitent aussi acheminés, le tunnel serait bénéfique au réseau.
- **Trou noir :** L'attaque par trou noir est très simple. C'est une attaque contre la retransmission de paquets. Il suffit au nœud attaquant de faire croire qu'il achemine les paquets en provenance d'un nœud source vers un nœud destination, et de ne pas le faire. Cette attaque n'est pas très subtile. En effet, une surcharge au niveau du nœud le rend facilement détectable et il sera ainsi déclaré malicieux. **Trou gris :** Cette attaque se base exactement sur la même idée que l'attaque de trou noir, à la différence près que cette dernière est un peu plus subtile. En effet, le trou malicieux ne va pas s'abstenir de transmettre tous les paquets, il va transmettre une partie et arrêter une autre. Ce comportement est plus difficilement repérable car il fait croire à une perte normale des paquets et rend donc l'attaque plus efficace.
- **Drainage :** Le drainage (*sink hole*) est une attaque basée sur le détournement des flux afin de faire partie des routes d'une façon non légitime. Il y a plusieurs manières de faire du drainage, comme par la précipitation cité dans les attaques par dénis de service, ou encore l'attaque par trou de vers vu ci-dessus.
- **Vol d'identité :** Cette attaque connue sous le nom *spoofing attack* n'est pas spécifique aux RAHM. C'est une attaque classique sur le réseau LAN. Elle consiste en le vol d'identité d'un nœud existant en s'appropriant son adresse physique ou encore sa clé. Le nœud malicieux s'approprie donc l'identité d'un nœud légitime. Dans les RAHM, les identités ne sont généralement pas connues d'avance. Ceci rend encore plus difficile l'identification des nœuds. Un exemple de vol d'identité est l'attaque **sybil**. Dans [Newsome *et al.* (2004)], l'attaque sybil consiste en la génération de fausses identités, cela veut dire que l'attaquant se trouve en possession d'un ou de plusieurs nœuds virtuels et arrive à

compromettre d'autres nœuds par l'envoi de messages HELLO et VHELLO en leur faisant croire à l'existence d'un ou de plusieurs nœuds qui n'existent pas.

2.3 Les solutions de sécurisation proposées

Dans la littérature, il existe plusieurs tentatives utilisant différentes approches afin de résoudre les différents problèmes liés à la sécurité des RAHM. Dans cette section, nous allons essayer de passer en revue les plus intéressantes d'entre elles et d'expliquer leur fonctionnement, leurs forces et leurs faiblesses. Nous allons aussi les classifier par type, tout comme nous l'avons fait à la section précédente, au niveau des attaques.

2.3.1 Gestion de clés et de certificats

Comme déjà expliqué auparavant, les RAHM ne disposent pas d'infrastructure centralisée. Les applications basées sur l'authentification utilisant la cryptographie à clé publique et sont donc très difficiles à mettre en place. En effet, l'hypothèse de l'existence d'une autorité de certification centralisée n'est pas garantie, or c'est la signature de cette supposée autorité qui garantie qu'une clé publique donnée appartient bien à un propriétaire donné et non à un imposteur. Le contrôle de l'authenticité de la signature n'est pas tout. En effet, il est tout autant important de s'assurer de la validité du certificat et de vérifier qu'il n'a pas été révoqué et qu'il n'est pas périmé, car les certificats sont généralement émis avec une date de début et une date de fin. Ceci est indispensable contre le vol et la divulgation des clés. Dans la littérature, nous relevons trois différentes façons d'aborder l'authentification dans les RAHM. Deux d'entre elles sont basées sur l'établissement d'une clé secrète qui rend possible l'authentification des participants. Ces deux approches : Relation maître esclave (*The duckling security policy model*) et Clé secrète commune (*The key agreement*), doivent surmonter la complexité qui réside dans la façon d'établir la clé secrète. La troisième façon s'est débarrassée du besoin d'une entité centrale de certification en se basent sur la cryptographie à clé publique. Cette approche est nommée Infrastructure à clé publique auto organisée.

2.3.1.1 Relation maître-esclave (*The duckling security policy model*)

Dans [Stajano et Anderson (1999)], les auteurs font des prévisions pour un futur où les objets de la vie courante seront tous dotés de possibilités de communication, de capacité de calcul, ...etc. Le début de la communication entre ces objets se fait en mode maître-esclave, via un canal radio ou autre. Un objet qui vient de joindre le réseau pour la première fois se voit marqué par son propriétaire d'un sceau (*imprinting*). Ceci permet de débuter l'échange d'une clé secrète entre le nouvel objet et un autre déjà présent sur le réseau. La procédure fait la supposition que le canal de communication est sûr, sans préciser comment. Cela peut être fait par un contact physique ou par un canal radio sécurisé par exemple. L'inconvénient de cette approche est que la gestion des clés se fait par une autorité centrale. Celle-ci s'occupe de lister les objets. Cette centralisation, même si elle semble simplifier le rôle des nœuds quant à la gestion de clés, n'est pas très réaliste car aucun nœud central n'est prévu dans les RAHM. Il est possible de donner cette charge à un nœud du réseau, mais ceci implique le risque que ce nœud ne soit pas digne de confiance et qu'il divulgue ces informations. Et même dans le cas où le nœud joue le jeu, il reste toujours le risque qu'il se fasse attaquer et devienne donc compromis. Cela met sur la table l'inévitable problème de la vie privée.

2.3.1.2 Clé secrète commune (*The key agreement*)

Dans la littérature traitant des protocoles à clé secrète commune, tout le monde s'accorde à dire que le plus difficile se résume dans l'établissement d'une clé commune entre plusieurs participants censés ne pas se connaître. Les nœuds participants se forgent et se partagent alors une clé secrète permettant leur authentification mutuelle. Ils disposent ainsi d'un moyen de communication sécurisé. Il y a deux façons permettant la mise en place d'une telle clé, d'une manière distribuée ou par contribution des participants.

- **Par distribution :** Dans ce cas de figure, l'hypothèse de l'existence d'un canal sûr doit être faite, car la distribution de la clé se fait via ce canal. Un exemple pratique serait des collègues réunis dans une salle close se distribuant un mot de passe inscrit sur un bout de papier. Aucune personne en dehors de la salle n'a connaissance de cette clé. La distribution de clé trouve ses limites dans l'existence d'un canal sûr. Cette hypothèse n'est pas très réaliste dans le cadre

des RAHM, et l'exemple des collègues de bureau ressemble d'ailleurs plus à un réseau filaire fixe.

- **Méthode Diffie-Hellman :** Alice et Bob se mettent d'accord sur un entier N et un générateur α du groupe cyclique fini d'ordre N (ce groupe est constitué de tous les entiers positifs ou nul strictement inférieurs à N. Les calculs dans le groupe cyclique se font modulo N). Alice et Bob choisissent chacun un nombre secret utilisé comme exposant. Le secret d'Alice est a, et celui de Bob est b. Alice envoie alors $\alpha^a modulo(N)$ à Bob et Bob envoie $\alpha^b modulo(N)$ à Alice :

$$Alice \longrightarrow Bob : \alpha^a modulo(N)$$
$$Bob \longrightarrow Alice : \alpha^b modulo(N)$$

Une fois que Bob a reçu $\alpha^a modulo(N)$ de la part d'Alice, il peut utiliser son nombre secret b pour calculer :

$$(\alpha^a modulo(N))^b modulo(N) \Longrightarrow (\alpha^a)^b modulo(N) \Longrightarrow (\alpha^{a.b}) modulo(N)$$

et Alice peut de son côté calculer :

$$(\alpha^b modulo(N))^a modulo(N) \Longrightarrow (\alpha^b)^a modulo(N) \Longrightarrow (\alpha^{b.a}) modulo(N)$$

La clé qui en résulte et qui est le secret partagé par Alice et Bob, est donc :

$$\alpha^{a.b} modulo(N)$$

Une tierce personne qui pourrait écouter les communications entre Alice et Bob, n'a aucune possibilité de savoir ce qu'ils se disent, car elle devrait deviner la clé secrète. Cette tâche est vraiment ardue, voir impossible dans un temps fini et avec une capacité de calcul finie si, bien sûr, N et a ainsi que l'exposant sont assez grands. Calculer a à partir de N et de α « revient à calculer un logarithme discret dans le groupe cyclique fini d'ordre N ». L'algorithme Diffie-Hellman ainsi présenté reste vulnérable à une attaque du type *man in the middle* et doit aussi être généralisé à un nombre d'utilisateurs supérieure à deux. Ceci n'est pas une tâche facile, non pas en terme de l'algorithme en soit, mais en terme de son application et des inconvénients accompagnant celle-ci. Chaque nœud n conserve son secret e^n et la clé commune devient $\alpha^{e_1.e2_2.e_3...e_n}$. L'envoi des α^{e_i} par chaque nœud aux autres nœuds n'est, au fait, pas suffisant,car il faut que le ième nœud reçoive $\alpha^{e_1.e2_2....e_{i-1}.e_{i+1}...e_n}$ pour pouvoir calcuer la clé. Pour fonctionner correctement un ordre entre les nœuds doit être

préalablement établi. En plus, l'avant dernier nœud et son prédécesseur, c'est-à-dire l'avant avant dernier nœud, doivent communiquer avec tous les autres nœuds du réseau, ce qui engendre des contraintes très difficiles à satisfaire pour un RAHM.

2.3.1.3 Infrastructure à clé publique auto organisée

Pour cette approche, les auteurs dans [Hubaux *et al.* (2001)], ainsi que dans [Capkun *et al.* (2003)] s'affranchissent de l'handicap de la centralisation. Cette façon de faire ne demande aucune infrastructure préétablie, ce qui adhère plus au concept des réseaux ad hoc mobiles. Basé sur des infrastructures à clé publique (*public key ifrastructure*) géré par les nœuds du réseau, ce modèle s'inspire beaucoup de PGP. En effet, chaque nœud met en place des certificats pour les nœuds en qui il a confiance. Lorsqu'un nœud A rencontre un nœud B, ils s'échangent leurs clés et leurs certificats via un canal sécurisé. Les nœuds étant mobiles, ces rencontres devraient être assez fréquentes. Dans l'éventualité où le nœud A veut communiquer avec le nœud B, si A a déjà le certificat de B (c'est-à-dire A connait B) la communication peut se faire une fois l'authentification établie. Sinon, ils passent à la comparaison de leur liste de certificats, afin de trouver un nœud commun C, dans lequel tous les deux ont confiance. Une chaîne de confiance est ainsi créée. Ceci permet aux nœuds qui ne se sont jamais rencontrés de communiquer, à condition d'avoir un ami commun, en l'occurrence le nœud C. Cette chaîne ainsi établie ne peut, dans la pratique, s'allonger indéfiniment, d'ailleurs sa longueur est volontairement limitée à trois. Ceci a pour avantage de réduire le nombre de certificats qu'il faut s'échanger. Ce protocole, même s'il semble assez adapté à la réalité des RAHM, reste pas assez efficace, car même après une période assez longue (plus que 3 minutes) les nœuds n'auront accès qu'à moins que la moitié des certificats et clés des nœuds présents sur le réseau. L'autre faiblesse vient du fait que la vérification de l'identité d'un tiers est laissée aux soins des nœuds, or rien ne garantit que ces derniers le font adéquatement.

2.3.1.4 Cryptographie à seuil

Dans [Zhou et Haas (1999)], l'idée derrière cette approche est d'éviter la compromission d'un nœud central qui aurait la tâche de distribuer les clés et les certificats, et de se protéger contre la situation où un ou plusieurs attaquants le prennent pour cible. Même s'ils n'arrivent pas à le corrompre, ils peuvent l'attaquer par dénis de

service. Dans ce cas, tout le réseau sera paralysé. Avec cette méthode, il y a un nombre n de nœuds qui ont la tâche de distribuer des certificats. Pour qu'un certificat soit valide, il faut qu'un nombre $t+1$ de nœuds participent à sa création, avec $t < n$. Dans ce cas, pour qu'une attaque marche, il faut que l'attaquant puisse avoir au moins $t+1$ nœuds à sa disposition.

2.3.2 L'authentification

L'authentification est nécessaire dans tout protocole qui se respecte. Elle est généralement basée sur la cryptographie à clés asymétriques comme RSA par exemple. Ce type de cryptographie n'est pas vraiment adapté à la réalité des réseaux Ad-hoc mobiles. Cela demande une grande capacité de calcul et les nœuds des RAHM n'en disposent généralement pas. Pour cela, des protocoles basés sur les chaînes de hachage ont vu le jour. Ces derniers sont de loin, moins gourmands en terme de capacité de calcul. Le plus cité dans la littérature est TESLA [Perrig *et al.* (2002)].

2.3.2.1 TESLA

Ce protocole d'authentification se base sur un code d'authentification de message utilisant des clés produites par chaînes de hachage à sens unique, tel que SHA-1 ou encore MD5. Ces derniers produisent à partir d'une chaîne de bits d'une longueur quelconque une chaîne d'une longueur bien déterminée. Cette chaîne est unique car il n'y a pas de collision. Les clés ainsi produites sont relâchées sur le réseau en suivant un échéancier très précis connu par tous les participants,
À cet effet, l'émetteur d'un paquet partage le temps en plusieurs intervalles égaux de même taille T. Il forme, par la suite, une chaîne de hachage d'une longueur L, constituée de plusieurs éléments h. Ces éléments sont associés, dans un ordre inverse, aux intervalles de temps. Les clés sont formées grâce aux éléments de la chaîne de hachage et une fonction pseudo aléatoire F. L'émetteur doit faire la distribution par un canal sécuritaire. Les paquets reçus par le récepteur sont mémorisés en attendant la réception de l'élément de hachage qui a servi à la génération de la clé. Cela permet de savoir si le paquet provient bien de son émetteur supposé.
Ceci assure une authentification pas chère en coût de calcul mais oblige les nœuds à être parfaitement synchronisés et d'avoir une capacité de stockage suffisante pour mémoriser les messages avant de les authentifier. Ceci génère aussi un délai supplémentaire

nécessaire à l'authentification.

2.3.3 Le routage

Les réseaux Ad-hoc mobiles nécessitent leurs propres protocoles de routage. Ces derniers doivent relever plusieurs défits de sécurité, ce qui a donné naissance aux protocoles de routage sécurisés. ARIADNE [Hu *et al.* (2002)] est parmi les plus cités dans la littérature.

2.3.3.1 ARIADNE

Basé sur le protocole de routage DSR (*Dynamic Source Routing*), ARIADNE est un protocole de routage sur demande qui peut utiliser différents protocoles d'authentifications existants, comme TESLA ou RSA. Il protége la formation des tables de routage, le champ de l'entête *nombre de saut* et assure l'intégrité, Il ne protége par contre pas contre les trous de vers ou les attaques par précipitation. Il fonctionne comme suit :
Le paquet de requête de route (*Route Request*) suit le chemin entre la source et la destination. Chaque nœud ainsi traversé, ajoute son adresse et signe ce paquet. Une fois la destination atteinte, le paquet emprunte le même chemin pour son voyage de retour et traverse ainsi les mêmes nœuds, ce qui assure la non modification des champs. Si une route n'est plus valable, pour une raison ou une autre, un paquet de notification est envoyé aux autres nœuds. Le fonctionnement de ARIADNE nécessite le déploiement et l'échange des clés et des certificats avant sa mise en marche, mais aucune description de cette phase, ni comment elle devrait être réalisée, n'est citée par les auteurs. Le protocole nécessite aussi des liens bidirectionnels entre les nœuds pour assurer son fonctionnement, ce qui n'est toujours pas le cas dans les réseaux Ad-hoc mobiles. Ceci pourrait, par exemple venir du fait que les antennes ont des portés différentes.
Malgré tout, ARIADNE reste un protocole assez léger et assez efficace en terme de consommation de ressources. Il assure une bonne sécurité.

2.3.3.2 Protection contre la précipitation

Les protocoles de routage sécuritaires rencontrés ne protégent pas contre les attaques par précipitation comme le *Rushing attack*. Dans [Hu *et al.* (2003b)] les au-

teurs affirment qu'il n'y a aucun protocole de routage sécuritaire capable de lutter contre le *Rushing attack* et proposent un mécanisme permettant de lutter contre ce type d'attaques assurant ainsi une meilleure sécurité. Ce mécanisme porte le nom de *Rushing attack Prevention* (RAP). Il est assez flexible et peut donc être ajouté aux différents protocoles de routage sécuritaire comme ARIADNE. Même si RAP est dédié à la protection contre le *Rushing attack*, il ne prévient pas les attaques avec certitude, mais le fait avec une certaine probabilité. Dans [Hu *et al.* (2003b)] cette probabilité est de $(\frac{n-m}{n})^l$, ou n représente le nombre de chemins existants entre le nœud source et le nœud destination incluant les chemins non légitimes, m représente le nombre de nœuds légitimes et l le nombre de sauts qu'il y a entre le nœud source et le nœud destination. L'application de ce protocole permet, malgré tout, d'éviter le blocage subit par les protocoles de routage sécuritaires qui les paralyse quand ils subissent une *Rushing attack*. Certains inconvénients sont à signaler : sous attaque, les délais pour trouver une route seront très affectés et une certaine latence due aux multiples essaies avant de trouver une route valable se fait ressentir. Cette latence existe même quand il n'y a aucune attaque. Elle dégrade sérieusement la qualité de service quand le trafic sur le réseau est assez important. D'autre part, tous les liens sur le réseau devraient être bidirectionnels. Les liens unidirectionnels seront ignorés et ne seront donc pas exploités. Tous les nœuds du réseau doivent être dotés d'un mécanisme qui synchronise leurs horloges, sinon RAP ne pourra pas fonctionner.
Le fonctionnement de RAP se base sur les deux principes suivants :

- S'assurer que son voisin prétendu est un vrai voisin par le calcul de la distance les séparant. Ceci est fait grâce au temps mis par le paquet de sollicitation de voisinage pour revenir et grâce à la vitesse de transmission du medium qui n'est autre que la célérité de la lumière. Il faut aussi s'assurer que tous les autres nœuds font de même afin d'assurer la route. Toute réponse dépassant un certain délai est considérée malicieuse. Il est à noter que dans ce cas, un nœud surchargé qui met plus de temps à répondre sera considéré comme voisin non légitime.
- Acheminer aléatoirement les requêtes en attendant la réception de plusieurs mêmes requêtes de différentes instances et, choisir aléatoirement une d'entre elles.

2.3.3.3 Protection contre les trous de vers

L'idée permettant de lutter efficacement contre les attaques par trous de vers est d'empêcher le voyage d'un paquet sur une distance D supérieure à d, distance égale à la portée habituelle d'un nœud du réseau. Ceci permet de détecter et ainsi d'éliminer tout paquet ayant emprunté un tunnel. Ce type de solution ne fait aucune différence entre un tunnel légitime et un tunnel malicieux et se contente de les éliminer.
Afin de pouvoir cerner les distances parcourues par les paquets, deux approches sont proposées dans [Hu *et al.* (2003a)]. La première est basée sur le temps, la deuxième sur la position.

Protection par la position Ce mécanisme exploite les positions des nœuds afin de déterminer la distance parcourue par un paquet et de l'empêcher ainsi de voyager à travers un trou de ver. À cet effet, chaque paquet doit être signé par son nœud émetteur et doit contenir sa position *pem* et le temps *tem* auquel il a été émis. À son arrivé à destination, le paquet doit subir une vérification par le nœud qui le reçoit. Elle consiste à savoir si le paquet a, oui ou non, dépassé la distance normale d. Il utilise à cette fin la formule suivante : $d > \| pem - pre \| + 2v*(tre + tem) + \delta$ avec *per* la position du nœud récepteur, v vitesse maximale de déplacement d'un nœud, *tre* le temps auquel le paquet est reçu et δ l'erreur maximale de la position d'un nœud. Si cette formule n'est pas vérifiée, le paquet sera rejeté, autrement il est accepté.

Protection par le temps L'idée derrière la protection par le temps est la même que pour la protection par la position, à savoir empêcher les paquets de voyager sur de très grandes distances. C'est juste le moyen qui diffère. En effet, c'est le temps et non pas la position qui détermine la distance parcourue par un paquet. Une grandeur $texp = tem + d/C - \Delta$ est alors introduite. C'est le temps après lequel un paquet devrait expirer où *tem* est le temps auquel le paquet est emis, d la distance égale à la portée habituelle d'un nœud du réseau, C la célérité de la lumière et Δ l'erreur maximale de la synchronisation entre deux horloges de deux nœuds différents. Si le paquet arrive à destination à un temps inférieur au *texp* il est accepté sinon, le paquet est rejeté.
Ces deux mécanismes présentent beaucoup d'inconvenients. Ils nécessitent le déploiement d'équipements supplémentaires, dont les nœuds du réseau n'en sont pas forcement dotés. En effet, pour connaître sa position avec exactitude un nœud a par exemple

besoin d'un GPS, car le nœud est mobile et change constamment de position. Les auteurs ne parlent pas de comment sont synchronisées les horloges et font des suppositions quant à leurs précision, ce qui n'est pas très réaliste. D'autre part, les nœuds sont obligés de dévoiler leurs positions ce qui pourrait permettre à un attaquant de profiter de ces informations.

2.3.3.4 Protection par le Multi chemins

Une autre technique de protection par le routage est l'envoi des données par plusieurs chemins. Même si l'idée derrière cette technique semble simple, l'application elle ne l'est pas. En effet, il faut être capable de trouver les n chemins que vont emprunter les données parmi le N chemins existant dans le réseau et reliant la source à la destination. Il faut aussi avoir un protocole qui supporte ce type d'envoi et trouver la bonne façon de diviser les données.

Dans [Bouam et Othman (2003)] les auteurs utilisent un protocole appelé SDMP

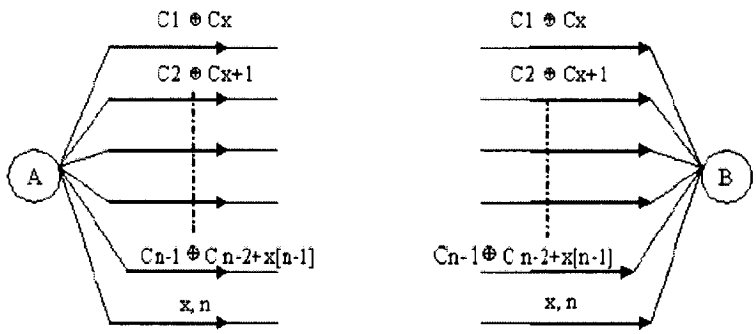


FIGURE 2.1 Opération XOR Bouam et Othman (2003)

(*Secured Data based Multipath Protocol*). Pour fonctionner correctement, le SDMP a besoin de prendre en considération la topologie du réseau, car s'il y a moins de 3 chemins différents entre le nœud source A et le nœud destination B, le protocole ne peut plus s'appliquer. La nécessité des 3 chemins s'explique par le fait que ce protocole utilise un canal dédié à la signalisation. Alors pour faire de l'envoi multichemins, il faut, au moins, deux chemins pour les données et un chemin pour la signalisation d'où le nombre 3 ou plus. Avant son envoi le message et donc divisé en $(n-1)$ parties préalablement identifiées. Suite à quoi un entier x avec$(1 < x \leq (n-1))$ est généré aléatoirement. Cet entier x ainsi que le nombre n de chemins seront envoyés sur le

lien de signalisation. Les $(n-1)$ parts du message sont ensuite combinées grâce à une opération XOR relativement à l'entier x. Cette opération d'XOR consiste à combiner les parts du message selon l'algorithme illustré à la Figure 2.1, tirée directement de [Bouam et Othman (2003)]. Chaque combinaison est par la suite, envoyée sur l'un des $(n-1)$ chemins et la $x^{ième}$ partie ainsi que l'entier x et le nombre n de chemins sont envoyés en clair. Ils serviront comme point de départ permettant la reconstitution du message initial.
La technique de l'envoi multichemins vise à augmenter la robustesse des réseaux Ad-hoc mobiles ainsi que la sécurisation des données transitant dessus. Elle oblige l'attaquant à assurer une présence massive sur le réseau ce qui nécessite un très grand déploiement de moyens et peut être fortement dissuasif. Dans [Bouam et Othman (2003)], quelques faiblesses sont décelées. Le protocole cherche des chemins différents mais ne vérifie pas si les chemins sont disjoints. En effet, il suffit à l'attaquant d'être sur la partie commune de la route pour tout intercepter. D'autres part, le protocole utilise les n meilleurs chemins parmi les N chemins existant entre un nœud source et nœud destination, ce qui pourrait permettre à un attaquant de se placer, par une simple prédiction, sur ces routes. Dans leurs études de performances, les auteurs se sont contentés d'un simple modèle client serveur, ce qui n'est pas très réaliste.

2.3.4 Coopération et réputation

Les solutions proposées ci-dessus, à savoir les solutions pour un routage sécuritaire ou une gestion efficace de l'authentification, résolvent certes un grand nombre de problèmes liés à la sécurité dans les réseaux Ad-hoc mobiles. Mais passent à coté de l'un des plus gros problèmes : la coopération. Au début de ce chapitre, nous avons parlé des contraintes spécifiques aux RAHM, et parmi ces contraintes nous avons évoqué le comportement égoïste de certains nœuds du réseau, qui est généralement motivé par un souci d'économie de leur ressources énergétiques limitées. Ce comportement, qui à première vue, ne semble pas dérangeant, est en effet l'un des plus gros problèmes de sécurité dans les RAHM. Il peut, dans certaines situations, provoquer une paralysie totale du réseau. Il existe plusieurs solutions encourageant la coopération dans les RAHM. Ces solutions se basent sur la gestion de la réputation. L'idée de la réputation est d'allouer une cotation aux nœuds. Chaque fois qu'un nœud relaie les paquets des autres nœuds, il reçoit une cote positive. Chaque fois où il re-

fuse de le faire, il reçoit une cote négative. Si la cote d'un nœuds descend au dessous d'un certain seuil, on lui alloue une mauvaise réputation. Il est alors banni par les autres nœuds et se retrouve isolé. Ceci encourage les nœuds à la coopération. Parmi les solutions les plus citées dans la littérature on retrouve CORE et CONFIDANT.

2.3.4.1 CONFIDANT

Le protocole CONFIDANT(*Cooperation Of Nodes - Fairness In Dynamic Ad-hoc NeTworks*) [Buchegger et Boudec (2002a)], [Buchegger et Boudec (2002b)], est un protocole qui encourage la coopération par l'utilisation de la réputation. Son fonctionnement est basé sur quatre mécanismes implantés dans tous les nœuds du réseaux. Ces mécanismes sont le moniteur, le gestionnaire de confiance, le système de gestion des réputations et le gestionnaire de routes.

Le moniteur Même si ce mécanisme est présent dans tous les nœuds du réseaux, seuls les nœuds voisins directs d'un nœud malicieux sont capables de détecter les anomalies liées au comportement de ce dernier. Ces anomalies sont.

- Le non acheminement de paquets.
- Le drainage du trafic.
- Le routage sans erreur.
- La mise à jour trop fréquente des routes.
- L'acheminement non conforme des paquets.

Chaque fois que le moniteur remarque une anomalie dans le comportement d'un nœud, il la reporte au système de gestion des réputations.

Le gestionnaire de confiance Le gestionnaire de confiance est composé d'une table qui maintient l'historique des messages ALARM, d'une table contenant les confiances allouées aux nœuds du réseau, ainsi que d'une liste dans laquelle se trouvent les amis auxquels les messages ALARM seront envoyés. La principale fonction de ce mécanisme est la gestion des massages ALARM qui proviennent du nœud abritant le moniteur ou des moniteurs des autres nœuds, et ceci en entrée ou en sortie. Les messages ALARM signalent les nœuds fautifs ainsi que la nature des fautes commises par ces derniers. Cette fonction principale se décompose en sous-fonctions :

- Le calcul de la confiance allouée aux autres nœuds.
- La gestion de la table qui contient les confiances calculées.

- L'acheminement des messages ALARM.
- La filtration des messages ALARM suivant la confiance accordée au nœud qui les a émis.

Le système de gestion des réputations Comme son nom l'indique, le système de gestion des réputation a pour rôle la gestion de la table qui contient les réputations des différents nœuds dans le réseau. Sa tâche n'est, par contre, pas mince car ce mécanisme a pour responsabilité de faire la distinction entre les vraies et les fausses accusations qui proviennes des différents nœuds. Certaines accusations sont portées à tort par l'entremise de nœuds malicieux visant à nuire au réseau. Les auteurs ne donnent pas de détails sur la façon qui permet au système de vérifier le fondement de ces accusations.

Le gestionnaire de routes Une fois bâties, les tables et les données de la réputation doivent servir à prendre des mesures et des décisions. C'est le rôle du gestionnaire de routes. Ce mécanisme utilise les données recueillies sur la réputation afin de construire les routes en se basant sur des métriques de sécurité. Il détruit alors toute route passant par un ou plusieurs nœuds malicieux. Il doit aussi traiter toutes les requêtes qui ont été émises ou ont traversées un nœud malicieux. La nature de l'action à mener sur ces requêtes n'est pas précisée.

2.3.4.2 CORE

Tout comme le protocole CONFIDANT, le protocole CORE (*collaborative reputation*) est un protocole d'incitation à la coopération [Michiardi et Molva (2002)]. En plus de traiter les problèmes liés à l'égoïsme des nœuds, il traite aussi les problèmes des nœuds malicieux. Pour cela, les nœuds sont investis de la tâche de tenir un registre dédié à l'évaluation de la réputation des autres nœuds du réseau. Cette réputation est représentée par une valeur numérique comprise entre -1 et 1. Un nœud ayant la cote -1 est un nœud complètemment malicieux et un nœud ayant la cote 1 est un nœud qui a une conformité totale. À chaque fois qu'un nœud se conforme aux règles de routage, il voit sa cote de réputation augmenter jusqu'à atteindre la cote maximale de 1. En revanche, à chaque fois qu'un nœud se comporte d'une façon égoïste sans respecter les règles définies par le protocole, il voit sa cote de réputation diminuer jusqu'à atteindre la cote minimale de -1, ce qui le mène à une exclusion du

réseau. Au début, tout nœud inconnu se voit attribuer la cote de réputation zéro. Au démarrage du protocole, tous les nœuds commencent avec la cote zéro. Le protocole nécessite l'échange des cotes de réputation entres les nœuds du réseau. Seules les cotes de réputations positives sont échangées, ceci permet la mise en place d'un portrait du réseau et évite les attaques par dénis de service traduites par un envoie répétitif de cotes négatives qui vise à diminuer la réputation du nœud attaqué. Afin d'éviter la situation dans laquelle un nœud participe au réseau par intermittence, c'est à dire à chaque fois qu'il a besoin d'acheminer ses propres paquets, pour retomber tout de suite après en mode veille, la cote de réputation , si elle est positive, est périodiquement décrémenté.
Dans [Michiardi et Molva (2002)], les auteurs parlent de trois catégories de réputations : La réputation subjective, la réputation indirecte et la réputation fonctionnelle. Ces trois type de réputations sont combinées selon une certaine pondération pour donner la réputation globale donnée à un nœud.

La réputation subjective Cette réputation est dite subjective car elle est calculée selon les observations du nœud. Elle se base sur le observations présentes et les observations passées. Ces dernières écopent d'un pondération supérieure. Ceci a pour effet de valoriser l'historique d'un nœud et lui évite d'être trop vite pénalisé pour des pertes de paquets causées par le médium de transport par exemple.

La réputation indirecte Cette réputation porte bien son nom, car elle n'est pas directement calculée par le nœud, mais elle est récoltée chez les autres nœuds du réseau. Elle n'est que positive afin d'éviter la vengeance d'un nœud qui essayerait de ternir la réputation d'un autre nœud. Cette réputation s'incrémente à chaque fois qu'un nœud participe à l'acheminement d'un paquet et que ce dernier arrive à destination. L'information sur les nœuds ayant collaborés à la tâche se retrouve dans le paquet de retour.

La réputation fonctionnelle Grâce cette réputation, il est possible de calculer les réputations indirectes et subjectives, selon les observations menées sur le respect du protocole de routage ou l'acheminement des paquets.
Chaque nœud tient une table de réputation dans laquelle se trouve la cote de réputation de tous les autres nœuds du réseau. C'est une sorte de petite base de

données à quatre champs. Ce sont un identifiant unique pour chaque nœud, un certain nombre d'observations portant sur les récentes actions du nœud à coter, une liste contenant les réputation indirectes du nœud à coter et enfin la liste des réputations fonctionnelles. CORE utilise une entité appelée Watchdog qui a pour mission le discernement du bon ou du mauvais fonctionnement d'un nœud. Cette entité est sollicitée par les nœuds à chaque fois où il faut valider une opération effectuée par un de leurs nœuds voisins. Tout comme CONFIDANT, CORE traîne certaines faiblesses. Les auteurs font aussi la supposition que tous les liens sur le réseau sont bidirectionnels ce qui présente une faible hypothèse. D'autre part, tout lien malicieux ayant une cote de réputation négative pourrait déjouer le protocole en changeant son identité après une brève déconnexion et profite ainsi d'une réputation vierge. Les protocoles de réputation ne font pas la différence entre un nœud égoïste et un nœud en difficulté, comme un nœud subissant une attaque par dénis de service.

2.3.4.3 Les Nuglets

La réputation n'est pas la seule façon qui pourrait forcer la collaboration des nœuds dans les réseaux Ad-hoc mobiles. La théorie des jeux est une autre stratégie permettant d'atteindre le même but, soit faire collaborer les nœuds. La théorie des jeux est une stratégie qui modélise les actions des usagers jouant un même jeu et œuvrant chacun pour son propre intérêt. Ce jeu comporte des règles régissant les actions des usagers. Le but des usagers est de gagner le jeu. Un usager doit donc augmenter au maximum sa fonction d'utilité. Si cette fonction augmente l'usager gagne si, par contre, elle diminue l'usager perd. Appliquée aux réseaux Ad-hoc mobiles, la fonction d'utilité est une combinaison entre l'énergie d'un nœud donné et les ressources du réseau dont il profite. Un équilibre doit être atteint afin que les nœuds ne puissent gagner qu'en se conformant aux règles préétablies. Dans [Buttyán et Hubaux (2001)], les auteurs proposent l'usage d'une monnaie virtuelle appelée *nuglet*. Le but de l'utilisation de cette monnaie est de forcer la collaboration des nœuds du réseaux afin qu'ils acheminent les paquets des autres nœuds et de limiter leur appétit en terme de ressources réseau. Le principe de fonctionnement est le suivant : la monnaie est échangée entre les nœuds chaque fois qu'un nœud envoi ou reçoit du trafic. Un système de vente et d'achat, c'est à dire des gains et des dépenses, est donc établi. Un nœud va donc dépenser des *nuglets* lorsqu'il achemine des paquets et en gagner quand il en reçoit. L'idée derrière ce système de vente et d'achat est

que la bourse dont dispose chaque nœud est limitée, si un nœud tente de mener par exemple une attaque par dénis de service, il devra dépenser des *nuglets* sans en recevoir. Or ces dernières son limitées, et sans *nuglets*, un nœud est complètement paralysé. Il ne peut plus communiquer avec les autres nœuds.
Il existe deux différents concepts pour utiliser les *nuglets*. Le premier est le *Paquet Purse Model*, dans lequel toute la monnaie nécessaire à l'acheminement du paquet est incluse lors de son envoi. Une somme précise est déduite à chaque nœud traversé. Ceci nécessite une connaissance préalable du réseau afin de pouvoir inclure la somme exacte. Si le montant est inférieur au montant requis, le paquet et les *nuglets* investis seront perdu. Le second concept est le *Paquet Tade Model*. À la différence du premier concept, le nœud source vend son paquet au nœud voisin le plus offrant. Le nœud ayant acheté le paquet fait de même, et ainsi de suite jusqu'à atteindre le nœud destination. Dans ce cas l'investissement est à la charge du destinataire et non à l'expéditeur. Ce qui ne nécessite pas une connaissance préalable du réseau comme est le cas pour le premier concept. Ceci engendre cependant une faiblesse liée au fait qu'un nœud peut inonder le réseau avec ces paquets sans aucune dépense.
L'utilisation de la monnaie dans les RAHM présente plusieurs faiblesses. Les auteurs ne parlent pas de comment remplacer les investissements perdus, que ce soit à cause des pertes normales de paquets car dans un réseau il y a toujours des pertes, ou par une perte causée par une attaque destinée à ruiner un nœud donné. La distribution des *nuglets* se fait par carte, ceci voudrait dire que tous les nœuds devraient être équipés, ce qui limite l'accès au réseau et augmente les coûts qui s'y rattachent.

2.4 Avenues de recherche

Dans ce chapitre, nous avons essayé de faire le tour et l'évaluation des solutions existantes, afin de mieux cerner les problèmes entourant les réseaux Ad-hoc mobiles. Nous nous sommes fixé comme objectif de travailler sur la garantie de l'intégrité des données sur les réseaux Ad-hoc mobiles, car nous avons remarqué que même si plusieurs solutions rencontrées dans la littérature traitent de cet aspect de la sécurité, aucune ne se dévoue complètement à la garantie de l'intégrité des données. Parmi les solutions qui luttent contre les attaques sur l'intégrité, nous pouvons citer la cryptographie qui garantie l'intégrité mais aussi la confidentialité des données. Aussi, l'utilisation de la réputation dans les réseaux Ad-hoc mobiles ainsi que l'utilisation

de l'envoi multichemins. Les deux dernières solutions proposées nous ont semblées plus à même d'être étudiées, car elles présentent un grand potentiel de recherche et plusieurs défis à surmonter.

Utilisation de la réputation dans les réseaux Ad-hoc mobiles : Il existe plusieurs failles dans ce type de solutions qui laisse une large marge de manœuvre et plusieurs possibilités de recherches. L'exemple de la déconnexion d'un nœud malicieux ayant cumulé une mauvaise réputation pour se reconnecter avec une nouvelle identité vierge en fait partie.

Toutefois, quoique cette technique permette de mieux garantir l'intégrité des données, elle n'est pas vraiment dédiée à cette tâche. Il existe, d'autres parts, plusieurs travaux portant sur la réputation dans les réseaux, ce qui nous a mené à choisir de travailler sur la solution de l'envoi multichemins.

Utilisation de l'envoi multichemins : Cette technique nous semble assez prometteuse et ne jouit pas de l'intérêt qu'elle mérite dans la littérature. Elle est innovante et est plus adaptée à la garantie de l'intégrité des données. Elle permet de lutter contre les attaques venant de source unique mais, aussi contre les attaques utilisant la collaborations des nœuds malicieux. Elle semble, en outre plus adaptée aux conditions particulières des réseaux Ad-hoc mobiles et pourrait même tourner certains désavantages de ce type de réseaux à son profit.

CHAPITRE 3

Envoi multi-chemins disjoints pour l'intégrité des données

Comme nous l'avons décrit dans le Chapitre 2, nous avons essayé de faire le tour et l'évaluation des solutions existantes, afin de mieux cerner les problèmes entourant la sécurité des réseaux Ad-hoc mobiles tel que la disponibilité des services, la confidentialité des données et leur intégrité ainsi que les problèmes reliés à l'identité tel que l'authentification et la non répudiation. La sécurisation de ces réseaux soulève encore plusieurs défis. Certains d'entre eux ont été déjà relevés avec un succès plus ou moins relatif. À cet effet, différentes approches ont été tentées, mais il reste plusieurs autres problèmes à résoudre.

3.1 Présentation de la solution

Dans ce chapitre nous allons présenter une solution traitant un des problèmes classiques de la sécurité dans les réseaux Ad-hoc mobiles, à savoir le problème de la garantie de l'intégrité des données. Cette solution tente d'exploiter et de mettre à profit une des propriétés des réseaux Ad-hoc mobiles, traduite par l'existence de plusieurs chemins entre un nœud source et un nœud destination. Elle consiste à adapter un algorithme de routage utilisant plusieurs chemins complètement disjoints pour l'envoi des paquets. Les chemins sont utilisés par blocs et non pas tous en même temps. L'algorithme change de chemin chaque fois que celui-ci est perdu. Cette solution est inspirée de la combinaison de deux approches déjà existantes mais qui sont employées à des fins différentes. La première est l'utilisation de l'envoi multi-chemins afin de garantir la robustesse du protocole de routage face à la mobilité dans les réseaux Ad-hoc. La deuxième est l'utilisation des sauts de fréquence mise en œuvre par les militaires afin de lutter contre le dénis de service causé par le brouillage des ondes.

Tel que nous l'avons mentionné au Chapitre 2, l'utilisation de l'envoi multichemins pour améliorer la sécurité dans les réseaux Ad-hoc a déjà été proposée par [Bouam et Othman (2003)]. Cette solution ne tient cependant pas compte de l'importance de la disjonction des chemin trouvés ni de la mobilité des nœuds. En effet, il suffit à l'attaquant d'être sur la partie commune des routes utilisées pour tout intercepter. Notre solution proposée sera détaillée plus loin dans ce chapitre.

3.2 Terminologie et définitions

Avant de commencer l'explication détaillée du protocole, nous devons donner plus de précision sur le sens d'un certain nombre de termes que nous allons utiliser tout au long de ce chapitre tel que multi-chemins, chemins disjoints.

3.2.1 Multi-chemins

Le terme *multi-chemins* est utilisé pour signifier l'existence et l'utilisation de plusieurs chemins afin de faire communiquer deux nœuds sur un réseau. Il existe donc plus d'un chemin pouvant acheminer l'information entre un nœud source S et un nœud destination D. Même si cette propriété n'est pas toujours vérifiée sur les réseaux conventionnels, elle demeure fortement probable sur les réseaux Ad-hoc de par leur nature, voir figure 3.1.

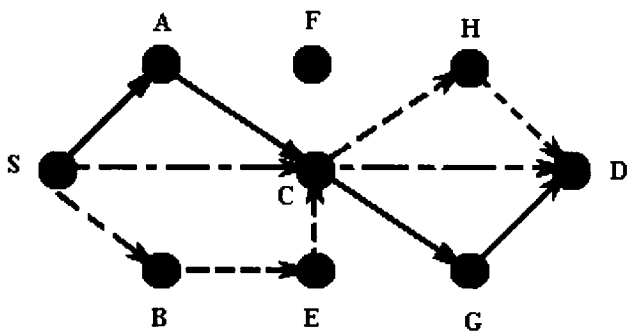


FIGURE 3.1 Multichemin - Illustration de trois chemins entre les nœuds S et D, *S-A-C-G-D, S-C-D et S-B-E-H-D*

3.2.2 Chemins disjoints

Le terme *chemin disjoint* est utilisé afin de signifier l'existence sur le réseau d'un lien multi-chemins entre un nœud source S et un nœud destination D. Ce lien multi-chemins a cependant la particularité suivante : Tous les liens le composant sont totalement séparés. C'est-à-dire qu'ils n'ont aucune arrête ni aucun nœud en commun et sont donc complètement disjoints, voir figure 3.2. Un lien multi-chemins

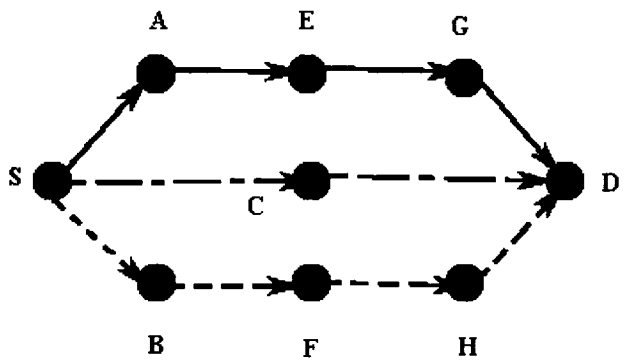


FIGURE 3.2 Chemins completement disjoints - Illustration de trois chemins entre les nœuds S et D, *S-A-E-G-D, S-C-D et S-B-F-H-D*

n'est pas forcément disjoint, c'est-à-dire qu'une ou plusieurs parties des différents chemins composant ce lien peuvent être communes. Un lien multi-chemins peut donc être totalement ou partiellement disjoint.

3.2.3 Chemins partiellement disjoints

Il existe, dans ce cas de figure, deux possibilités. La première possibilité est l'existence d'un ou de plusieurs nœuds communs entre les différents chemins constituant le lien multi-chemins entre les nœuds sources S et destination D. Mais il n'y a, par contre, aucune arrête commune entre les chemins. Ceci est illustré à la figure 3.3. Dans cette situation, C représente une nœud d'intersection et est donc commun aux trois chemins.

La deuxième possibilité se traduit par l'existence d'un ou de plusieurs nœuds ainsi que d'une ou de plusieurs arêtes en commun entre les différents chemins constituant

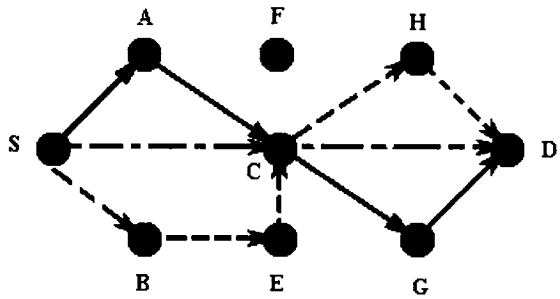


FIGURE 3.3 Chemins partiellement disjoints avec nœuds communs - Illustration de trois chemins entre les nœuds sources S et destination D, *S-A-C-G-D*, *S-C-D et S-B-E-H-D*

le lien multi-chemins entre les nœuds sources S et destination D. Ceci est illustré à lar figure 3.4. Nous remarquons que les nœuds E et G sont communs aux trois che-

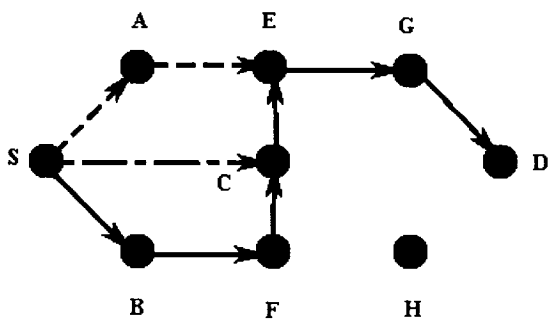


FIGURE 3.4 Chemins partiellement disjoints avec arrêtes et nœuds communs - Illustration de trois chemins entre les nœuds sources S et destination D, *S-B-F-C-E-G-D*, *S-C-E-G-D et S-A-E-G-D*

mins, et que le nœud C et communs aux chemins S-B-F-C-E-G-D, S-C-E-G-D. Les arrêtes E-G et G-D sont communes aux trois chemins, et l'arrête C-E est commune aux chemins S-B-F-C-E-G-D, S-C-E-G-D.

3.3 Principe de l'algorithme et fonctionnement

Dans cette section, nous allons définir et expliquer avec précision toutes les phases du déroulement du protocole *Secure Disjoint Multipath Routing Protocol* SDMRP.

3.3.1 Hypothèses

Avant d'entrer dans les détails du protocole, nous devons émettre les hypothèses et suppositions de départ suivantes :

1. Il existe dans les réseaux des routes disjointes reliant les nœuds source S et destination D .
2. Il n'y a pas de modifications apportées aux couches du protocole *TCP/IP*

3.3.2 Principe de SDMRP

Le protocole se base sur l'idée de l'envoi multi chemins. À cet effet, un mécanisme de découverte de routes entre le nœud source S et le nœud destination D est mis en marche. Ce mécanisme permet la découverte de routes ayant pour caractéristique d'être, dans la mesure du possible, disjointes.
Dans certains cas rares, l'existence de chemins disjoints entre les nœuds source S et destination D n'est pas garantie. Ceci est illustré à la figure 3.5.

Afin que le protocole puisse fonctionner, l'existence de deux routes au moins est nécessaire. Si le nombre n de routes est inférieure à deux ($n < 2$), le protocole AODV est employé. Ce cas de figure est assez rare, mais la topologie changeante d'un réseau en perpétuel mouvement, crée certains scénarios qui sont peu probables mais réalisables. Cette situation est illustrée dans la figure 3.6. Dans les réseaux de grande densité, cette situation est pratiquement improbable.
Si ($n \geq 2$), l'algorithme choisi n routes parmi les N routes trouvées avec ($2 \leq n \leq N$). Le message **M** à envoyer passe par un mécanisme de décomposition qui le divise en m parties appelées $\mathrm{h(Gp}_i)$. Chaque Gp_i est hachée en vue d'obtenir son empreinte. Les empreintes $\mathrm{h(Gp}_i)$ ainsi obtenues sont stockées en vue de leur envoi sur les différents chemins sélectionnés reliant le nœud sources S et le nœud destination D. À leur arrivée au nœud destination D, les empreintes $\mathrm{h(Gp}_i)$ ainsi que les parties Gp_i auquel elles sont associées passent par un mécanisme de vérification.

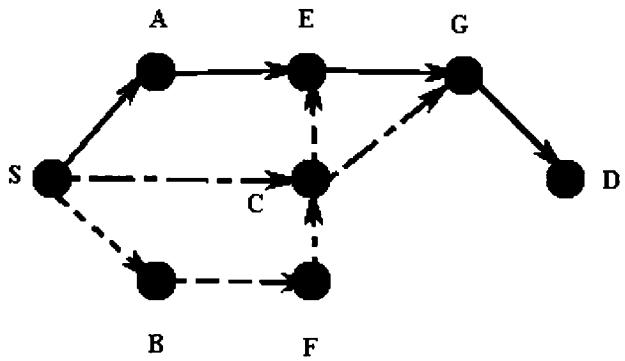


FIGURE 3.5 Les routes *S-A-E-G-D et S-B-F-C-E-G-D*, ont deux nœuds ainsi que deux liens communs, les nœuds E et G et les liens E-G et G-D

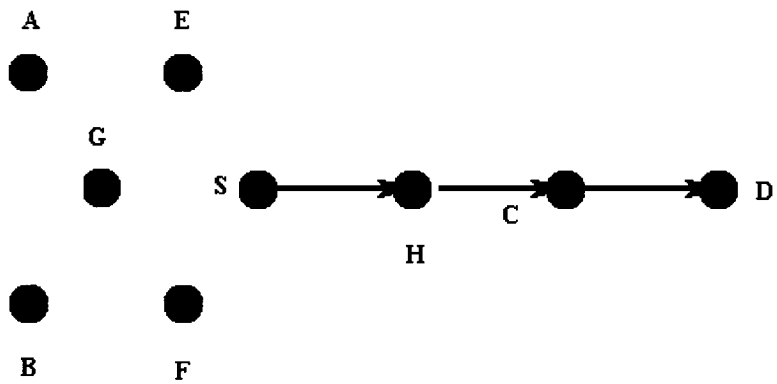


FIGURE 3.6 L'existance de plus d'une route n'est pas garantie - Existance d'une seule route reliant les nœuds source S et destination D, soit *S-H-C-D*

Ce mécanisme s'assure que toutes les empreintes $h(Gp_i)$ ayant transité par différents chemins correspondent bien au Gp_i auquel elles sont associées. Si une ou plusieurs empreintes ne correspondent pas à leur Gp_i ou une ou plusieurs différences entre les empreintes sont décelées, le Gp_i ainsi que ses empreintes sont rejetés. Ceci est dû au fait qu'à cette étape, nous ne sommes pas capables de déterminer l'endroit ou s'est opéré la ou les modifications et de discerner les bons des mauvais. Si cette situation se répète, il est impératif de rejeter toutes les routes et de les remplacer

par d'autres routes, que se soit en relançant une nouvelle recherche de routes ou en puisant dans le reste des N routes déjà à disposition, dépendamment du nombre de routes restantes. Ceci permet d'éviter de boucler et tomber ainsi dans le panneau d'une attaque par dénis de service. Une liste tabou contenant les anciennes routes à éviter serait utile pour ne pas retomber sur le ou les mêmes nœuds malicieux. Cette liste n'a pas été implémentée dans le cadre de ce travail. Si, par contre, les empreintes correspondent, le Gp_i est accepté. Un mécanisme de recomposition de message est prévu afin de reconstituer le message initial M à partir des différents groupes de paquet Gp. Ces étapes sont illustrées dans les figures 3.7 et 3.8.

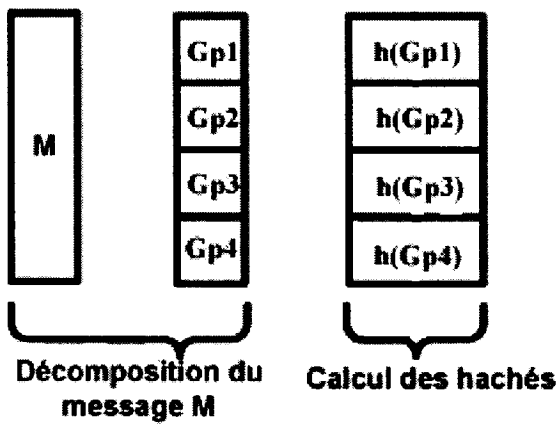


FIGURE 3.7 Décomposition et calcul des hachés du message M

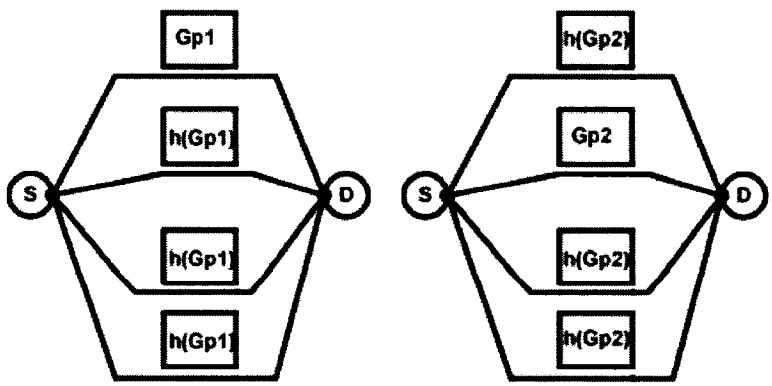


FIGURE 3.8 Envoi des groupes de paquets et de leur hachés

Il ne suffit pas à ce protocole de trouver les routes et de gérer l'envoi et la réception

des messages, car quand on parle des réseaux Ad-hoc mobiles, on parle de mobilité et donc de la gestion et du maintien des routes. Étant en perpétuel mouvement, les nœuds du réseau changent de position, agissant ainsi sur la topologie du réseau. Les routes déjà calculées ont donc une durée de vie limitée et dépendante de la mobilité des nœuds. La gestion et le maintien des routes est donc d'une importance extrême. Le fait que ce protocole se base sur l'utilisation et le maintien de plusieurs routes lui procure un certain avantage. Car si une route ou même plusieurs (dans de certaines proportions) sont perdues, les autres routes continuent à fonctionner. En effet, si parmi les N routes trouvées, une portion k de routes est perdue, tant que $(N-k \geq n)$ avec n le nombre minimal de routes fixées par le protocole et avec $n \geq 2$, le protocole continue à fonctionner. Dans le cas ou le nombre de routes encore valables descend sous la barre de n c'est-à-dire que $(N-k < n)$, une recherche de routes est initiée. Ceci procure une certaine robustesse vis-à-vis de la mobilité et limite la signalisation due à une recherche répétitive.

3.3.3 Algorithme

Afin de fonctionner correctement, l'algorithme principal du protocole SDMRP nécessite une entrée le renseignant sur la topologie du réseau. À cet effet, un deuxième algorithme de découverte de routes est indispensable. Celui-ci a pour tâche de déterminer les routes reliant les nœuds source S et destination D et de faire en sorte que les routes soient disjointes dans la mesure du possible. Quand il est appelé par l'algorithme principal, l'algorithme de découverte de routes initie sa recherche et retourne les routes reliant les nœuds source S et destination D, s'il en existe. Dans notre cas, nous avons choisi d'utiliser l'algorithme de découverte de routes utilisé dans le protocole *Ad-hoc On-demand Multipath Distance Vector routing* (AOMDV) développé par M. K. Marina et S. R. Das [2001].

3.3.3.1 Algorithme principal de SDMRP

Il est utile de définir la terminologie avant d'énoncer notre algorithme principal.

Soit :

N	Nombre de routes existantes entre les nœuds source S et destination D.
n	Nombre de routes effectivement utilisées entre les nœuds source S et destination D.
M	Message à transmettre entre les nœuds source S et destination D.
m	Nombre de portions du message M.
Gp_i $\{i = 1$ à $n\}$	$i^{ème}$ portion du message M.
$\mathrm{h}(\mathrm{Gp}_i)$	Le résultat du hachage de Gp_i.

Au début, le nœud source S émet l'intention de transmettre vers le nœud destination D, l'algorithme de découverte de route est donc appelé pour trouver le maximum de routes disjointes reliant les nœuds source et destination. L'algorithme vérifie que le nombre N de routes trouvées est supérieur à deux. Un nombre n de route est alors choisi parmi les N routes trouvées. Les données M à transmettre sont décomposées en m parties. Chaque partie forme un Gp_i. Les $\mathrm{h}(\mathrm{Gp}_i)$ sont calculés. Suite à quoi, les $\mathrm{h}(\mathrm{Gp}_i)$ sont stockés en vue de leur envoi. Un choix initial de route parmi les n routes sélectionnées est effectué. Les Gp_i sont envoyés sur la route choisie, et leurs $\mathrm{h}(\mathrm{Gp}_i)$ sur les routes restantes. Si une route devient défectueuse, elle est immédiatement remplacée par une autre route dans N, tant que N reste supérieure à n. Si N devient inférieure à n, l'algorithme de découverte de route est appelé à nouveau. Un pseudo code de cet algorithme est présenté à la figure 3.9.

Au niveau du nœud destination, dès que les Gp_i arrivent, ils sont hachés pour créer des $\mathrm{h}(\mathrm{Gp}_i)$ qui vont servir de point de comparaison et de validation pour les $\mathrm{h}(\mathrm{Gp}_i)$ ayant transité par le réseau.

Si les $\mathrm{h}(\mathrm{Gp}_i)$ correspondent à leurs homologues créés à partir du (Gp_i) qui vient d'arriver au nœud destination, le (Gp_i) en question est accepté et un accusé de réception est envoyé au nœud source. Sinon il est rejeté et une demande de renvoi est envoyée au nœud source.

```
initialiser nombre de route N = 0
N = Algodécouverte( )
tant que (données à transmettre)
  Si N ≥ 2 faire
     choisir n liens, n ≥ 2 dans N
     décomposer M pour former les Gp_i
     calculer h(Gp_i)
     tant que (Gp à transmettre)
        choisir la prochaine route dans n
        envoyer Gp_i
        envoyer h(Gp_i)sur les autres routes
        Si route défectueuse faire
              N = N - 1
              Si N ≥ n faire
                   remplacer route
              Sinon
                   N = Algodécouverte( )
                   choisir n liens, dans N
     fin tant que
  Sinon
     utiliser AODV
fin tant que
```

FIGURE 3.9 Pseudocode de l'Algorithme principal

3.3.3.2 Algorithme de découverte de routes

Comme son nom l'indique, cet algorithme est appelé par l'algorithme principal au début de son exécution afin de lui retourner les routes liées à la requête du nœud source pour atteindre le nœud destination. Cet algorithme est aussi appelé chaque fois que le nombre de routes restantes descend au dessous de la limite de deux routes puisque, tel qu'indiqué précédemment, le routage multichemins devient impossible avec moins de deux routes.

Développé par M. K. Marina et S. R. Das [2001] pour améliorer le comportement et les performances de AODV, dans le cadre d'un réseau Ad-hoc à grande mobilité, le protocole AOMDV maintient plus d'une route reliant le nœud source au nœud destination afin de pouvoir continuer à fonctionner même si la route utilisée devient hors d'usage.

L'algorithme puise donc une autre route dans la liste des routes qu'il maintient et change la route défectueuse par la nouvelle route encore valide. Les routes maintenues ont la particularité d'être disjointes afin qu'elle n'aient pas de partie commune qui risque de rendre non valables le restant des routes. Le fait que l'algorithme puise une route dans la liste qu'il maintient diminue, voir élimine, la latence due à la recherche d'une nouvelle route et évite l'envoi des requêtes qui inondent le réseau, engendrant ainsi un grand trafic de signalisation.
L'algorithme se compose de deux parties. La première consiste en une règle servant à la mise à jour des routes afin d'établir et de maintenir, à chaque nœud, des chemins sans boucles. La deuxième consiste en un protocole distribué servant à trouver des chemins n'ayant aucune arrête commune, et non pas complètement disjoint. Les auteurs ont choisi de faire ce compromis, car le nombre de chemins complètement disjoints est de loin inférieur au nombre de chemins sans arêtes communes.
Contrairement à [Das et Marina (2001)], ce compromis ne nous arrange pas, car notre priorité va à la sécurité, et non pas à la performance du protocole vis-à-vis de la grande mobilité. Une route ayant un ou plusieurs nœuds en commun avec une route déjà choisie doit être évitée au maximum, voir complètement proscrite, car c'est la présence des nœuds malicieux sur une route donnée que nous voulons absolument éviter. Même si deux routes sont différentes, il suffit qu'elles ait un nœud en commun, et que ce nœud soit malicieux, pour être aussi dangereuses l'une que l'autre. Le fait d'échanger la première route pour la deuxième, ne résoud donc pas notre problème dans une pareille situation.
Même si nous l'avons modifié pour être utilisé d'une manière différente et pour des raisons autres que la grande mobilité, à savoir la sécurité, cet algorithme garde une grande partie de ses avantages.
Dans la figure 3.10, tirée de [Das et Marina (2001)], la deuxième copie du RREQ (*Route Request*) venant de B est supprimée au niveau du nœud intermédiaire I. Toutefois, deux copies de la première copie venant de A parviennent à la destination D. Le nœud D émettra une réponse pour les deux copies de la requête même si les deux contiennent le même premier saut commun. Les chemins de retour se croisent en I mais ils constitueront, tout de même, des routes à liens disjoints.

Contrairement à l'algorithme de [Das et Marina (2001)], nous avons choisi de trouver des routes n'ayant ni lien ni nœud commun. La figure 3.11 illustre la manière

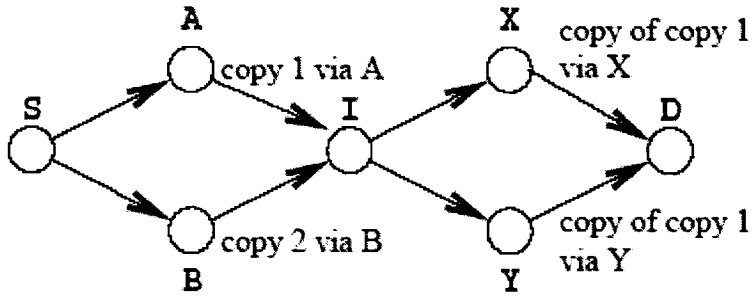


FIGURE 3.10 Découverte de routes à liens disjoints [Das et Marina (2001)]

avec laquelle l'algorithme procède. Le nœud source S émet un paquet RREQ sur tout le réseau. Les voisins directs du nœud S qui sont les nœuds A, F et B le reçoivent en premier. Le nœud B le transmet au nœud H, et les nœuds A et F le transmettent au nœud G. Le nœud G, quant à lui, ne transmet que le premier arrivé des deux retransmissions faite par le nœud A et le nœud F. Le paquet qui arrive en second est ignoré. Deux copies atteignent le nœud destination D via G et H. de cette façon nous obtenant deux routes disjointes entre les nœuds source S et destination D. La première est S-B-H-D et la deuxième dépendamment de qui est arrivé en premier à G, sera S-A-G-D ou bien S-F-G-D.

Le fait d'ignorer la duplication de la requête ayant emprunté un chemin différent et

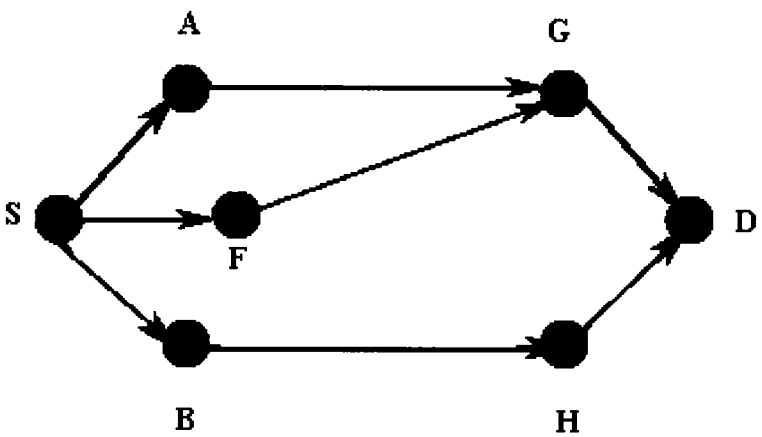


FIGURE 3.11 Découverte de routes à liens et nœud disjoints

arrivant en second, a une incidence qui va au delà de la garantie de la disjonction des routes. Cela permet aussi un choix local de la meilleure des deux routes, car plus

rapide que l'autre. Ce concept est illustré à la Figure 3.11.
Quoi qu'il ne garantit pas de trouver les meilleures routes reliant deux nœuds source et destination sur un réseau donné, tous comme AODV sur lequel il est basé, notre protocole permet de trouver des routes assez fiables, et assez rapides.
Certaines différences ont été apportées à la structure de la table de routage de AODV pour le rendre compatible avec le routage multichemin. Le champ *hopcount* est remplacé par le champ *advertised-hopcount* et le champ *nexthop* est remplacé par le champ *route-list*. Contrairement à l'entrée prochain saut (*nexthop*) de AODV, l'entrée *route-list* permet de définir plusieurs prochains sauts avec leur compteurs de saut (*hopcount*) respectifs. Toutefois les entrées prochain saut (*nexthop*) gardent leur même numéro de séquences de destination (*detination sequence number*). Le champ *advertised-hopcount* est initialisé à chaque mise à jour du champ *sequence number*. Les nœuds mettent à jour leur propres champs *advertised-hopcount* pour le nœud destination toutes les fois où ils envoient une annonce de route (*route advertisement*).
Le choix des routes revêt une très grande importance, car les paquets qui circulent sur des routes différentes n'arrivent pas forcément dans l'ordre voulu, à cause des différences qu'il y a sur les routes empruntées, qui n'ont ni les même longueurs ni les mêmes vitesses de transmission. Ceci a pour effet de devoir augmenter les tailles des tampons, et d'augmenter la latence avant de pouvoir reconstituer le message initial, d'où l'importance d'avoir une certaine homogénéité entre les routes choisies.
Dans la figure 3.12, nous avons illustré à titre d'exemple un réseau Ad-hoc sous une forme simplifiée. Sur ce réseau, trois routes différentes ayant la propriété d'être disjointes et reliant le nœud source S et le nœud destination D sont représentées. Ces routes diffèrent par leur longueur qui sont fonction du nombre de sauts entre les nœuds S et D, dans ce cas de figure les longueurs sont 11, 8 et 4. En faisant abstraction des vitesses de transmission des liens ainsi que de la charge des nœuds, on pourrait supposer que les charges des nœuds ainsi que leurs vitesses de transmission sont toutes égales. Ce sont donc les longueurs des routes qui fixent le temps que mettent les paquets entre les nœuds S et D. On suppose alors l'envoi de trois paquets, chacun sur une route différente. Celui qui emprunte la route le plus courte arrivera en premier même s'il n'a pas été envoyé en premier et ainsi de suite.
Soient les paquets P1, P2 et P3, ils sont envoyés séquentiellement et respectivement sur les routes de longueurs 11, 8 et 4. L'ordre d'arrivée au nœud destination D sera P3,

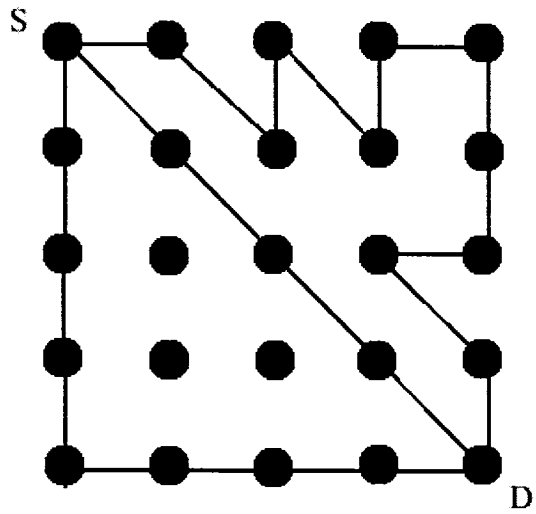


FIGURE 3.12 Importance de l'homogénéité des routes

P2 puis P1. Cette situation, qui a l'air anodine, pourrait poser de sérieux problèmes en terme de latence et peut rendre énorme la taille des tampons. Ce problème prend plus d'ampleur dans notre cas que dans le cas des autres applications, car dans notre protocole l'algorithme garantissant l'intégrité des données ne pourra accepter un paquet avant d'avoir effectué toutes les vérifications. Or ces vérifications nécessitent le recueil de tous les $h(Gp_i)$. Si l'un d'entre eux tarde à venir, le (Gp_i) déjà arrivé restera dans le tampon, jusqu'à l'arrivé de ce dernier, ce qui permettra d'effectuer les vérifications nécessaires et d'accepter le groupe ou de le rejeter. Afin d'éviter un temps d'attente infini et de nettoyer les tampons, le (Gp_i) est automatiquement rejeté après le dépassement du temps *timeout* fixé. Suite à quoi, une demande de retransmission du groupe rejeté est envoyée.

3.4 Étude théorique

Pour justifier le développement de notre protocole de routage multichemins sécurisé, ainsi que de déterminer son comportement face aux attaques qu'il risque de subir, il est nécessaire de mener une étude théorique. À cet effet il est impératif d'émettre un certain nombre d'hypothèses, et de vérifier théoriquement le comportement du protocole face à différentes situations.

3.4.1 Hypothèses de départ

Afin de simplifier la situation et de pouvoir modéliser un réseau Ad-hoc mobile type, nous devons émettre certaines hypothèses concernant la nature du réseau, sa taille, sa densité, la mobilité de ses nœuds, etc.

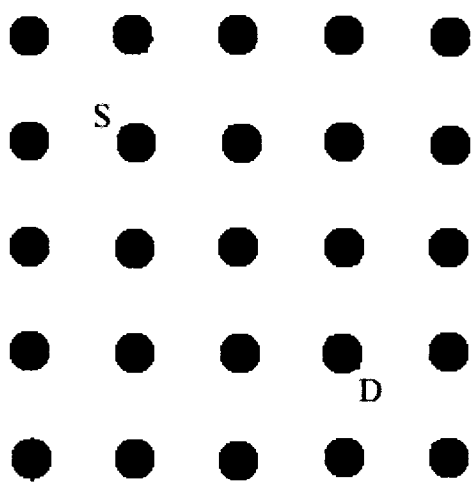


FIGURE 3.13 Matrice représentant la partie utile du réseau

Pour notre étude, nous allons considérer une partie du réseau qui inclue les nœuds source S et destination D ainsi que les chemins les reliant. Le reste du réseau ne sera pas représenté car il ne présente aucun intérêt en relation avec notre étude. À cet effet, nous allons représenter la portion du réseau par une matrice de cinq par cinq, c'est-à-dire un réseau composé de 25 nœuds dont le nœud source et le nœud destination. Voir figure 3.13.

1. *Hypothèse 1 :* Tous les nœuds du réseau sont munis d'antennes omnidirectionnelles leurs permettant d'atteindre et d'échanger avec tous leurs voisins immédiats distant d'un saut. La portée de ces antennes ne permet par contre pas d'atteindre les autres nœuds qui ne sont pas voisins directs et qui sont à une distance supérieure à un saut.
2. *Hypothèse 2 :* Tous les liens reliant les nœuds dans le réseau sont considérés comme liens bidirectionnels, sans quoi le protocole de routage ne pourra pas

fonctionner. À la base, AODV exige et nécessite des liens bidirectionnels pour assurer son fonctionnement.

3. *Hypothèse 3 :* Tous les nœuds du réseau peuvent être considérés suspects et malicieux, à l'exception des nœuds source S et destination D qui sont toujours considérés fiables et ne peuvent en aucun cas être corrompus.
4. *Hypothèse 4 :* Les nœuds considérés malicieux ont la possibilité d'employer deux types de stratégies : Ils peuvent agir seuls ou en collaboration avec d'autres nœuds malicieux du réseau.
5. *Hypothèse 5 :* Tout nœud du réseau considéré comme fiable peut basculer et devenir malicieux, résultant du fait qu'il vient d'être corrompu par un autre nœud malicieux ou que c'est une stratégie employée par ce nœud.
6. *Hypothèse 6 :* Les nœuds source S et destination D n'ont pas d'information leur permettant de distinguer les nœuds fiables des nœuds malicieux.
7. *Hypothèse 7 :* Pendant le laps de temps réduit de l'étude théorique, le réseau est considéré statique. Cette hypothèse se base sur le fait que la mobilité des nœuds est quasi nulle sur une observation instantanée du réseau.
8. *Hypothèse 8 :* Il n'y aura pas d'attaques portées contre le protocole de découverte de routes.

3.4.2 Situation 1 : pas de collaboration entre les nœuds

Dans les réseaux Ad-hoc mobiles, les problèmes reliés à la sécurité découlent généralement de la présence de nœuds malicieux au sein du réseau. Ces nœuds sont plus au moins dangereux et peuvent élaborer différentes stratégies d'attaques. En effet, les nœuds malicieux peuvent agir seuls ou en collaboration avec d'autres nœuds. Dans cette partie, nous avons choisi de commencer par l'étude de l'efficacité de notre protocole de routage multichemins face aux attaques sur l'intégrité des données placées par des nœuds malicieux agissant seuls sans aucune possible collaboration avec les autres nœuds du réseau. Nous allons aussi effectuer une comparaison entre notre protocole multichemins et un protocole de routage simple chemin face à ce type d'attaque sur l'intégrité des données.
Ce type d'attaque, c'est-à-dire sans collaboration, est plus facile à déployer, car la collaboration entre nœuds nécessite la présence de plus d'un nœud malicieux, et

exige un moyen de communication et une certaine coordination entre ces nœuds attaquants, ce qui s'avère plus difficile à réaliser.
Quoi que plus difficile à mettre en place et plus coûteux en terme de déploiement de moyens, ce type d'attaques est généralement plus efficace que les attaques individuelles. C'est la raison pour laquelle nous avons choisi de commencer par des attaques plus simples à déployer, mais aussi plus simples à contrer pour ensuite continuer avec celles qui sont plus difficiles à déployer mais aussi plus difficiles à contrer.

3.4.2.1 Protocole de routage conventionnel à une route

Dans ce cas de figure, un nœud malicieux E ayant de mauvaises intentions et cherchant le moyen d'intercepter les données qui transitent entre le nœud source S et le nœud destination D, dans le but de changer leur contenu, touchant ainsi à l'intégrité de ces données, devra forcement se trouver sur la route reliant les nœuds S et D. (voir figure 3.14)

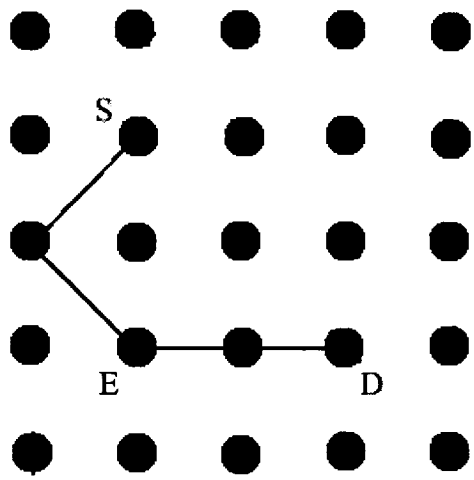


FIGURE 3.14 Présence d'un nœud malicieux E sur la route reliant S et D

Pour un nœud malicieux, faire partie de la route reliant les nœuds S et D afin de placer une attaque sur l'intégrité des données ne se fait heureusement pas à la

demande. En effet, le nœud attaquant ne peut pas prédire quels sont les nœuds qui veulent communiquer, ni quelle est la route qui les relie. Il doit donc compter sur sa chance pour faire partie d'une route donnée, et qui dit chance dit probabilité.
Calculons alors la probabilité pour qu'un nœud donné fasse partie d'une route en particulier sur le réseau. Cette probabilité sera alors égale à la probabilité qu'un message soit altéré durant son transit sur le réseau, après avoir quitté le nœud S et avant son arrivée à la destination D.

Soit :

r	La route reliant les nœuds source S et destination D.
l	La longueur de la route r qui est égale au nombre de ses nœuds sans compter la source S et la destination D.
p	La probabilité qu'un nœud x du réseau soit malicieux, cette probabilité est égale à la proportion des nœud malicieux sur le réseau, c'est-à-dire le quotient entre le nombre de nœuds malicieux et le nombre total des nœuds du réseau.
$(1-p)$	La probabilité qu'un nœud x ne soit pas malicieux.

La probabilité qu'aucun nœud de r ne soit malicieux est :

$$(1-p)^l \tag{3.1}$$

La probabilité qu'au moins un nœud de r soit malicieux et donc qu'une attaque est réussie est :

$$Pr(A = success) = 1 - (1-p)^l \tag{3.2}$$

L'intégrité des données transmises au moyen du protocole de routage conventionnel à une route est donc vulnérable à une attaque avec la probabilité calculée à l'équation 3.2.

3.4.2.2 Protocole de routage multichemins sécurisé

Dans ce deuxième cas de figure, un nœud malicieux E ayant l'intention d'intercepter les données transitant entre le nœud source S et le nœud destination D, afin

de changer leur contenu, s'attaquant ainsi à l'intégrité des données, devra forcément se trouver sur l'une des routes reliant les nœuds S et D, voir figure 3.15.

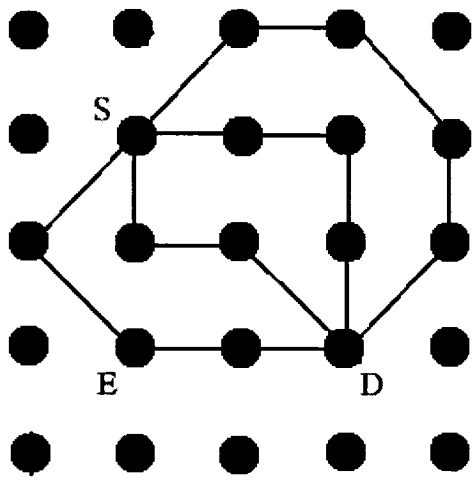


FIGURE 3.15 Présence d'un nœud malicieux E sur une des routes reliant S et D

Dans cette situation, le nœud malicieux E ne pourra en aucun cas réussir son attaque. Car comme nous l'avons précisé, même si le nœud E s'arrange pour être sur l'une des routes, son action sera très limitée, car les nœuds ne peuvent pas coopérer entre eux, ce qui rend la tâche de l'interception possible mais seulement d'une façon partielle. Un nœud malicieux E ne peut se trouver que sur une route à la fois car les routes sont disjointes. Ceci fait qu'au pire des cas une seule partie du message peut donc être interceptée pour être ensuite modifiée. Cette modification sera détectée au niveau du nœud destination D, les groupes de paquets Gp_i et leurs hachés ne transitant pas par les mêmes routes . Cette partie Gp du message ainsi que ses hachés $h(Gp_i)$ seront donc rejetés et retransmis via une autre route. L'attaquant aura, au pire, réussi à faire retransmettre une partie du message, mais en aucun cas pu modifier le contenu du message sans se faire détecter. Il est à noter que si l'intention de l'attaque était de produire un dénis de service, nous pouvons la considérer comme réussie. Pour contrer ce type d'attaque, il serait possible de fixer un seuil qui permet d'accepter un (Gp_i) si la majorité des $h(Gp_i)$ correspondent et non pas tous les $h(Gp_i)$.

En faisant baisser ce seuil, nous rendons plus difficile le succès d'une attaque par dénis de service, mais nous baissons la confiance que nous pouvons octroyer à l'intégrité des données.

3.4.3 Situation 2 : collaboration entre les nœuds

Dans ce cas de figure, nous faisons la supposition que n'importe quel nœud malicieux a la possibilité de collaborer avec n'importe quel autre nœud malicieux se trouvant sur le réseau. Cette collaboration n'est pas limitée par le nombre des nœuds qui y participent, elle peut donc s'effectuer à deux ou à plusieurs. Nous faisons aussi abstraction des moyens dont disposent ces nœuds afin de collaborer, et admettons que cette collaboration est possible en tout temps et sans aucune restriction.
Afin de mener à bien cette étude nous allons considérer le pire cas, se traduisant par la collaboration de tous les nœuds malicieux du réseau.

3.4.3.1 Protocole de routage conventionnel à une route

Pour le routage conventionnel à une route, la collaboration des nœuds ne change rien à la donne. Le fait que tout le message transite par une seule route fait en sorte que la présence d'un seul nœud malicieux sur cette route suffit à modifier le message sans être détecté par la destination. Il n'y a donc nul besoin de collaboration. Le calcul fait dans la section précédente reste donc valable.
La probabilité qu'au moins un nœud de la route r soit malicieux et donc qu'une attaque est réussie est : $Pr(A = success) = 1 - (1 - p)^l$.
C'est la probabilité que l'intégrité des données transmises au moyen du protocole de routage conventionnel à une route, soit vulnérable à une attaque non détectée.

3.4.3.2 Protocole de routage multichemins sécurisé

Pour pouvoir placer une attaque sur l'intégrité des données et modifier le contenu du message sans être détecté, l'attaquant devra forcement être présent sur toutes les routes en même temps. Le (Gp_i) et ses hachés h(Gp_i), passent chacun sur un chemin différent, c'est la stratégie du protocole multichemins. Il suffit que l'une des routes empruntées soit saine pour détecter les changements opérés sur les autres routes. Pour réussir, cette attaque exige la coordination et collaboration des nœuds attaquants ainsi que la présence d'au moins un nœud malicieux par route.

Pour simplifier les calculs nous allons considérer égale la longueur l des routes choisies. Dans la figure 3.16 l'attaque réussie où échoue selon s'il y a au moins un nœud

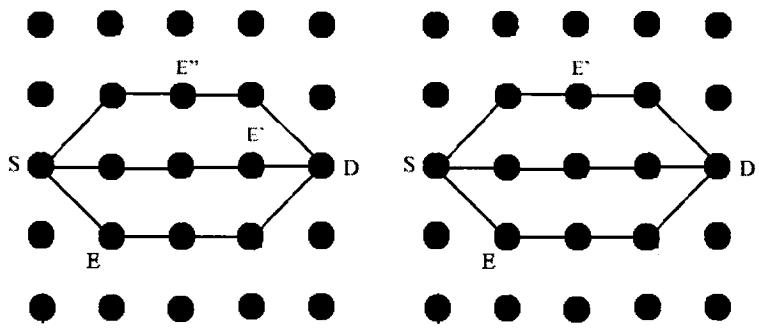


FIGURE 3.16 Attaque réussie et attaque échouée

malicieux sur chaque route utilisée.

La probabilité qu'une attaque soit réussie $Pr(A = Success)$ est donc égale à la probabilité qu'il y ait au moins un nœud malicieux sur chacune des routes choisies pour transmettre le message. Nous avons déjà calculé dans l'équation 3.2 la probabilité qu'au moins un nœud malicieux est présent sur une route de longueur l. Cette probabilité est de $1 - (1 - p)^l$. Pour n routes disjointes de longueur l, on aura :

$$Pr(A = Success) = (1 - (1 - p)^l)^n \tag{3.3}$$

Nous remarquons que nous avons rendu beaucoup plus ardue la tâche de l'attaquant. Ce dernier devra déployer de très grandes ressources pour pouvoir réussir son attaque à chaque fois que nous choisissons de rajouter une route supplémentaire, alors que de notre part nous n'avons pratiquement pas fourni d'effort. La probabilité de succès décroit donc exponentiellement par rapport au nombre de routes utilisées et le nombre de machines à compromettre augmente exponentiellement.

Plus le nombre de routes augmente plus l'adversaire est contraint d'augmenter son investissement en terme de présence sur le réseau, et ce en terme de moyens assurant la collaboration entre les nœuds qu'il détient, ce qui représente une tâche non négligeable. Nous illustrons l'évolution de cette probabilité de succès dans la Figure 3.17 pour une probabilité $p = 0,2$ qu'un noeud soit malicieux. Toutefois l'augmentation abusive du nombre de routes n'est pas gratuite. Elle est d'ailleurs limitée par le

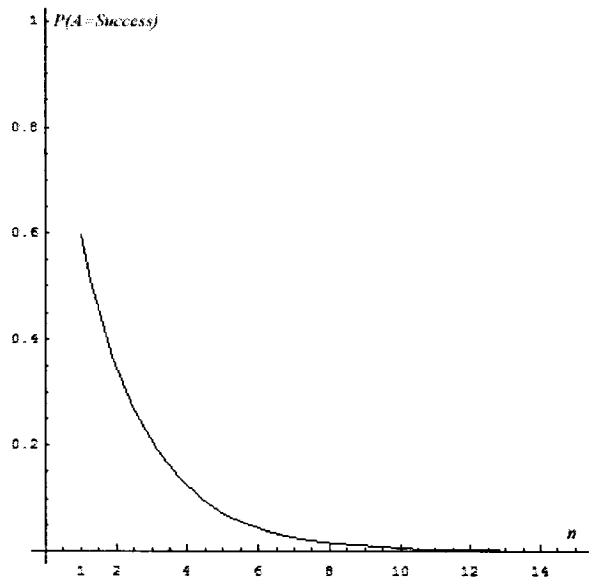


FIGURE 3.17 Évolution de la probabilté de succès d'une attaque en fonction de n

nombre réel de routes disjointes reliant la source S à la destination D. D'ailleurs, plus le nombre de routes est grand, plus grande est la perte en homogénéité des routes, ce qui a de sérieuses implications, en terme de délai et de performance du protocole. Un compromis doit donc être établi. Les pertes engendrées par la multiplication des routes et leurs implications seront étudiées ultérieurement.

3.4.4 Impact de la multiplication des routes

Les routes disjointes entre les nœuds source S et destination D, calculées par le protocole de routage présentent un certain nombre de différences. Les longueurs des routes en nombre de sauts et la qualité des liens en terme de charge des nœuds ainsi que de leurs vitesses de transmission sont les plus importantes et sont celles qui ont le plus d'impact sur le comportement du protocole.
Les Gp_i ainsi que leurs hachés $h(Gp_i)$ transitent par de différentes routes. Même si les Gp et leurs $h(Gp_i)$ sont envoyés en même temps, les différences de longueurs, de vitesses et de charges des routes empruntées font en sorte que ces derniers n'arrivent pas à destination en même temps. Or pour accepter un Gp_i, le protocole doit procéder à sa comparaison avec tous ses $h(Gp_i)$ ayant transités par toutes les routes utilisées. Tant que un $h(Gp_i)$ n'est pas encore arrivé, le (Gp_i) reste en attente et ne sera

pas validé. Cette contrainte nécessaire à la garantie de l'intégrité des paquets, est lourde de conséquences, car plus le nombre de routes est grand, plus les routes sont hétérogènes. Ceci augmente le temps nécessaire pour l'acceptation d'un Gp_i ainsi que la taille des tampons dans lesquels il réside durant cette attente. Ce problème de latence n'est pas seulement imputé à l'hétérogénéité des routes mais aussi à leur nombre comme nous allons l'expliquer plus tard dans ce chapitre.

3.4.4.1 Calcul de la latence

Pour pouvoir estimer la perte en qualité de service due à la multiplication des routes utilisées entre les nœuds source et destination, trois données techniques doivent être prises en compte. La latence, la variation de la latence appelée gigue ainsi que la perte de paquets. Nous allons procéder au calcul de la latence dans le cas d'une application conventionnelle simple chemin, ainsi que dans le cas d'une application mutichemins. Nous procéderons ensuite à la comparaison des deux pour déterminer la perte induite par la multiplication des routes. Dans la suite de ce travail nous considérons :

- *Hypothèse :* La latence sur un chemin donné d'un nœud source S à un nœud destination D suit une distribution normale d'une moyenne μ et d'une variance σ^2.

Cas de la route unique Sur une route unique, le total de la latence est exprimé suivant la loi normale décrite ci-haut.

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \quad e^{-(\frac{x-\mu)^2}{2\sigma^2})} \tag{3.4}$$

La probabilité que la latence soit inférieure à une valeur t est :

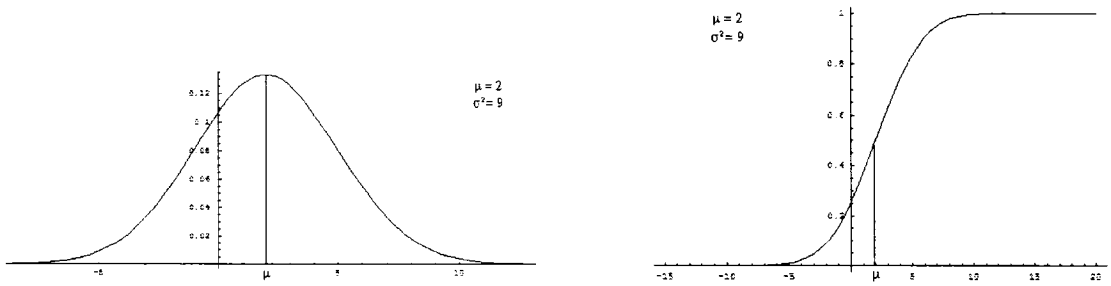
$$Pr(x \leq t) = CDF(x) = \int_{\infty}^{t} f(x, \mu, \sigma) \quad dx \tag{3.5}$$

$$Pr(x \leq t) = \frac{1}{2}(1 + erf(\frac{t-\mu}{\sigma\sqrt{2}})) \tag{3.6}$$

avec erf la fonction d'erreur de Gauss

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_{0}^{x} e^{-t^2} \quad dt \tag{3.7}$$

Nous illustrons dans les figures 3.18(a) et 3.18(b) la densité de probabilité *pdf* de la latence ainsi que sa fonction de répartition *CDF* pour une loi normale de moyenne $\mu = 2$ et de variance $\sigma^2 = 9$.



(a) PDF de la latence (b) CDF de la latence

FIGURE 3.18

Cas de plusieurs routes Dans notre modèle, la latence sur chacune des routes suit la loi normale décrite dans l'équation 3.4. Les exigences de notre protocole de routage multichemins nous obligent à considérer la latence sur les n routes utilisées. Les latences sur les routes sont indépendantes les unes des autres car les chemins sont complètement disjoints. Nous pouvons alors les considérer comme des événements indépendants.

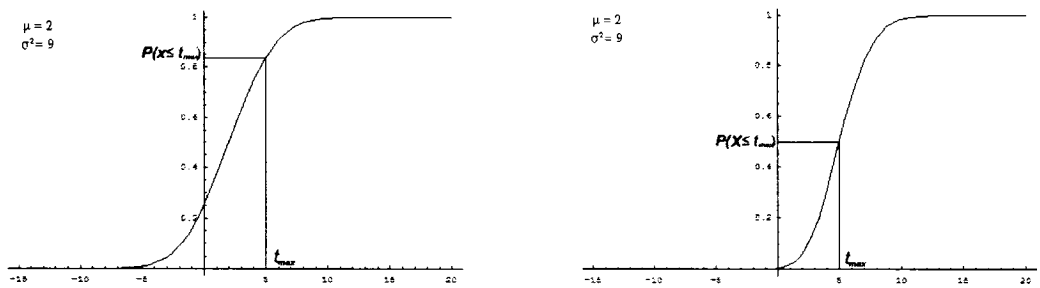
Afin d'estimer la latence permise sur les n routes, qui devra être inférieure à un seuil maximum, nous calculerons la probabilité que toutes les latences soient inférieures à un t_{max}. Les événements étant indépendants, cette probabilité est donc le produit des probabilités que la latence sur chaque route soit inférieure à t_{max}. Soit donc Y la variable aléatoire modélisant le produit des latences. Nous avons donc :

$$Pr(Y \leq t_{max}) = P(x_1 \leq t_{max})\ P(x_2 \leq t_{max}) \cdots P(x_n \leq t_{max}) \tag{3.8}$$

Afin de simplifier les calculs, nous faisons l'hypothèse que les latences sur les n routes utilisées suivent toutes la même loi avec le même μ et le même σ, ce qui nous permet d'écrire :

$$Pr(Y \leq t_{max}) = (\frac{1}{2}(1 + erf(\frac{y-\mu}{\sigma\sqrt{2}})))^n \tag{3.9}$$

Nous constatons que la probabilité que la latence soit inférieure à t_{max} diminue exponentiellement avec le nombre n de routes. Ceci est illustré par les figures 3.19(a) et 3.19(b) pour un nombre de routes $n = 1$ et $n = 4$ avec $\mu = 2$ et $\sigma^2 = 9$.



(a) probabilté pour une route (b) probabilté pour quatre routes

FIGURE 3.19

Nous remarquons donc que sur une seule route $n = 1$, la probabilité que la latence soit inférieur à une valeur $t_{max} = 5$ est $Pr(Y \leq 5) = 0.84$ tandis que pour $n = 4$ $Pr(Y \leq 5) = 0.5$.

Estimation de la moyenne et de la variance de la nouvelle distribution : Afin de pouvoir estimer la moyenne et la variance de la distribution de probabilité de la latence sur n routes, nous procédons au calcul de la densité de probabilité relative à $Pr(Y \leq t_{max})$. Nous dérivons donc l'équation 3.9 pour obtenir la fonction de densité suivante :

$$f_{max}(y, n, \mu, \sigma) = \frac{n\sqrt{2}\ e^{-\frac{(y-\mu)^2}{2\sigma^2}}(1 + erf(\frac{y-\mu}{\sigma\sqrt{2}}))^{n-1}}{2^n\sqrt{\mu}\ \sigma} \tag{3.10}$$

Nous illustrons dans la figure 3.20 les différentes fonctions de densité de probabilité pour $n = 1$, $n = 4$ et $n = 10$ routes. Nous pouvons remarquer que ces dernières perdent leur propriété de distribution normale au fur et à mesure que nous augmentons le nombre de routes.

Nous avons, par la suite, tracé pour une latence suivant initialement une loi normale $f(x, \mu, \sigma)$, l'évolution de la moyenne ainsi que l'évolution de la variance pour un nombre n de routes allant de $n = 1$ à $n = 30$.

Nous remarquons que la moyenne de la latence augmente avec l'augmentation du nombre de routes tel que illustré à la Figure 3.21. Ceci traduit une dégradation de la qualité dans le réseau. Toutefois, cette dégradation n'est pas exponentielle en n

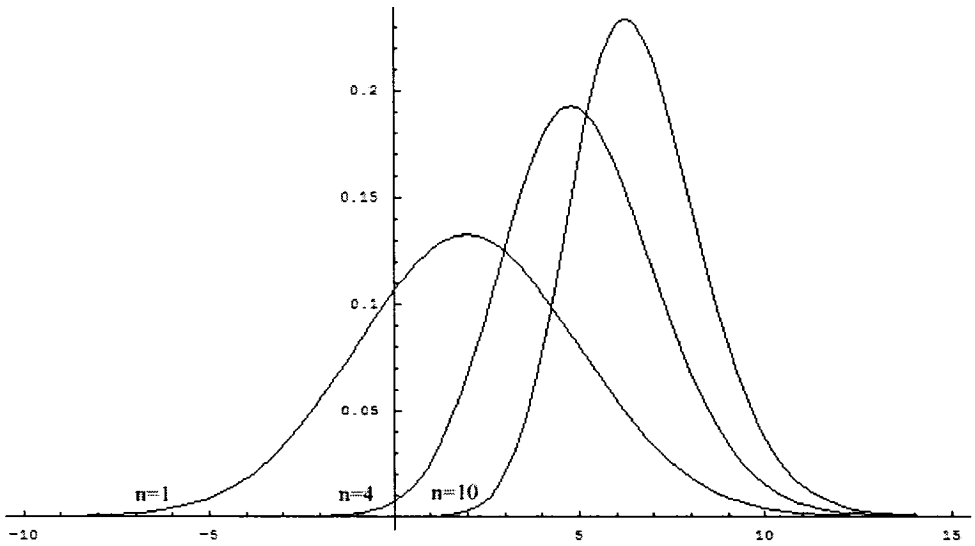


FIGURE 3.20 Densités de probabilité pour une, quatre et dix routes

contrairement à l'effort qu'un attaquant devra déployer pour compromettre au moins un nœud sur chaque route choisie, pour réussir son attaque (voir Figure3.17).
La variance, quant à elle, a tendance à diminuer avec l'augmentation du nombre de routes n. Ceci traduit une diminution de la dispersion des événements. La latence est donc resserrée autour de la latence moyenne donc la gigue diminue.
Le fait d'utiliser de l'envoi multichemins n'influence pas le taux de retransmission de paquet. C'est l'utilisation de la fonction de vérification de l'intégrité qui l'influence indirectement. Chaque fois qu'un (Gp_i) est rejeté à cause de la non concordance des h(Gp_i) due à un ou plusieurs nœuds malicieux, il y a retransmission du (Gp_i) ainsi que de tous ses h(Gp_i). Pour un algorithme ayant une tolérance zéro et n'utilisant pas de seuil permettant un compromis entre la garantie de l'intégrité des données et la sensibilité aux attaques par dénis de service, la probabilité de succès d'une attaque par dénis de service est :
Pour un algorithme mutichemin $Pr(DOS = succes) = 1 - (1 - p)^{ln}$
Pour un algorithme standard $Pr(DOS = succes) = 1 - (1 - p)^{l}$
avec :

- p est la probabilité qu'un nœud soit malicieux.
- l est la longueur d'une route.
- n est le nombre de routes utilisées.

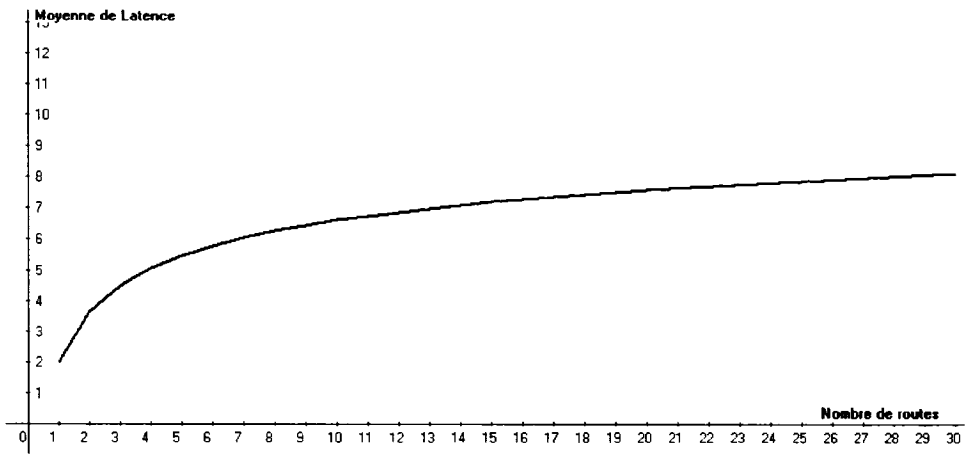


FIGURE 3.21 Évolution de la moyenne de la latence en fonction de n

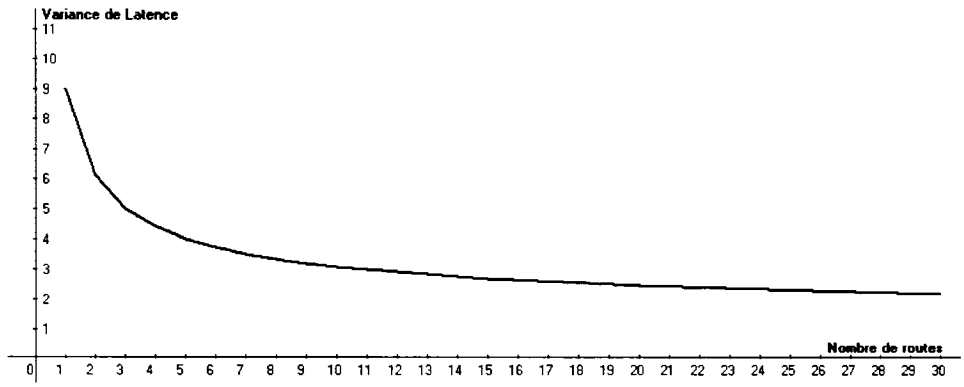


FIGURE 3.22 Évolution de la variance de la latence en fonction de n

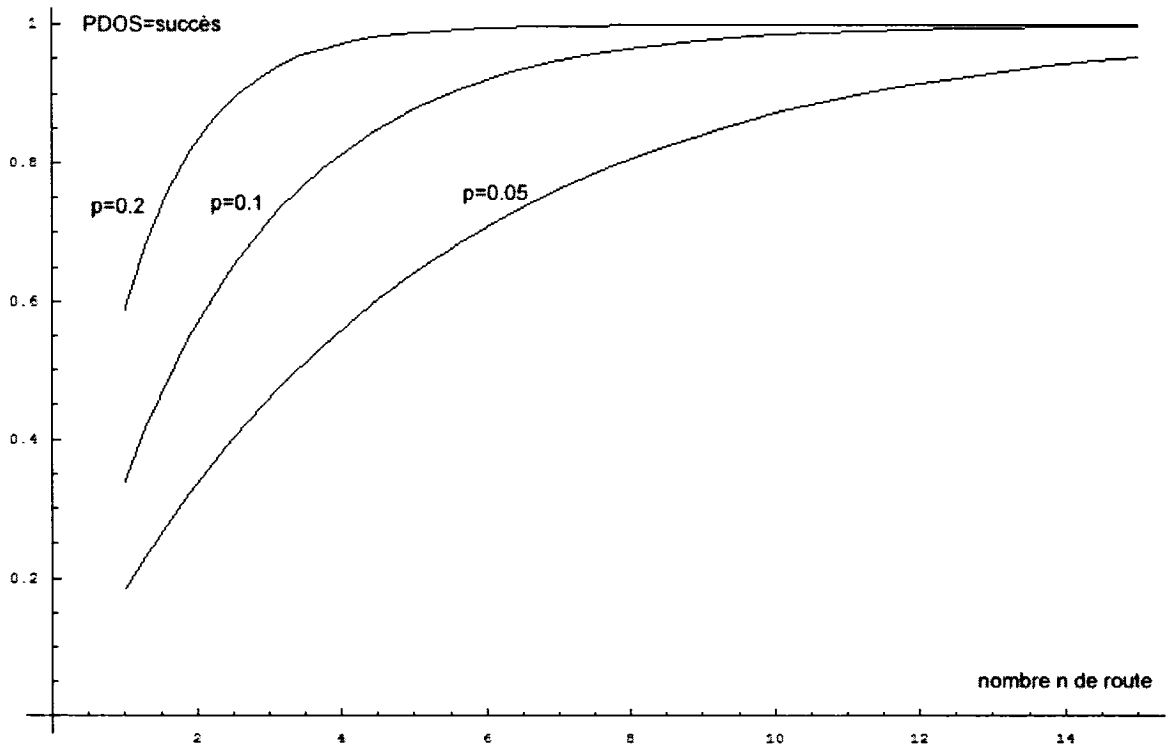


FIGURE 3.23 Probabilité de succès pour une attaque par dénis de service

CHAPITRE 4

Implémentation et Résultats

Dans ce chapitre, nous allons procéder à une évaluation des performances du protocole de routage multichemins que nous avons présenté dans le chapitre précédent. Contrairement au travail d'évaluat-ion analytique effectué au chapitre 3, cette évaluation sera pratique, elle se basera sur la simulation. À cet effet, nous allons commencer par la présentation du simulateur `ns-2` qui servira comme base de test pratique, suite à quoi nous allons présenter les détails de l'implémentation ainsi que le plan de tests et les métriques utilisées, ce qui nous permettra de vérifier et de valider les résultats escomptés et énoncés précédemment.

4.1 Environnement

Dans cette section nous allons donner une brève description des outils utilisés afin de réaliser nos tests, tel que la plate-forme, le simulateur...etc.

4.1.1 Plate-forme

Pour réaliser nos testes nous avons utilisé une plate-forme de type PC ayant les caractéristiques suivantes :

- Processeur Intel Pentium IV de 3.06 GHz
- Mémoire vive 512 Mo
- Système d'exploitation Linux Redhat 7.3

4.1.2 Le simulateur

Dans le cadre de notre recherche, nous avons fait le choix d'utiliser et d'intégrer une partie du travail réalisé par [Das et Marina (2001)] qui ont développé un algorithme permettant de modifier AODV afin qu'il puisse rechercher et maintenir plusieurs routes disjointes reliant un nœud source à un nœud destination pour des

raisons autre que la sécurité. À cet effet, nous avons récupéré une partie du code développé par [Das et Marina (2001)] auquel nous avons porté quelques modifications. Ceci afin de l'adapter pour qu'il puisse répondre à nos exigences. Pour pouvoir utiliser le code développé par [Das et Marina (2001)], nous avons du garder le même simulateur qu'ils ont utilisé à savoir le `ns-2`. Ce simulateur a été développé à l'origine dans le cadre du projet VINT par l'université de Berkeley en Californie en collaboration avec le DARPA ainsi que XEROX, c'est un simulateur réseau à événements discrets orienté objet.

Deux langages de programmation sont mis à l'œuvre pour développer ce simulateur l'incontournable `C++` et le langage de scripting `tcl`. Tous les modules sur lesquels se base le simulateur ainsi que les protocoles sont développés en `C++`. Le langage tcl fourni une interface permettant une certaine flexibilité qui facilite l'utilisation du simulateur. Cependant, pour ajouter ou intégrer un nouveau protocole le langage `C++` est indispensable.

Afin de réaliser nos simulations qui portent sur les réseaux Ad-hoc mobiles, certaines composantes reliées à ce type de réseau ne sont pas implémentées sur la version de base du simulateur `ns-2`. À cet effet, nous avons du ajouter les composantes réseau *Wireless and Mobility Extensions to ns-2* du groupe de recherche Monarch[1] développée pour la simulation des réseaux sans fils mobiles en supportant la couche physique, la couche liaison de données ainsi que le modèle MAC dans `ns-2`. La fonction de distribution *(DCF)* du IEEE 802.11 pour les réseaux locaux sans fils est utilisée comme couche MAC, le modèlel radio utilise les caractéristiques des interfaces radio commerciales ayant un débit nominal de 2 Mb/s et une porté de 250 mètres.

Le modèle radio permet l'utilisation de canaux sans fils avec ou sans erreurs, ceci donne une meilleure flexibilité permettant de mieux isoler certains phénomènes lors du déroulement des simulations. Le simulateur implémente une version récente des spécifications du modèle AODV. Ce modèle a servi de base au modèle AOMDV de [Das et Marina (2001)] que nous avons intégré et modifié pour les fins de notre recherche.

L'extension *Wireless and Mobility Extensions to ns-2*, a donc introduit le concept du nœud mobile par la création de la classe *(mobile node)*, voir figure 4.1. Elle a aussi permis de créer les stations de base ainsi que la prise en compte de la mobilité

[1]Site web : http ://www.monarch.cs.rice.edu/cmu-ns.html

des nœuds, cette extension inclue la plupart des protocoles de routage Ad-hoc et particulièrement AODV sur lequel s'est porté notre intérêt.

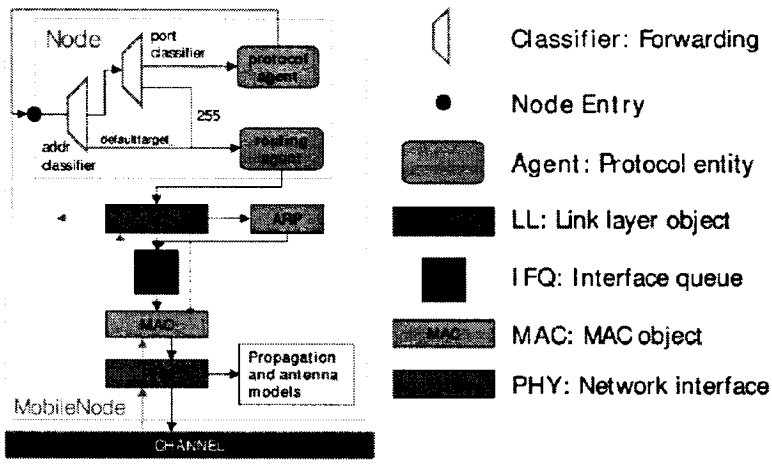


FIGURE 4.1 Structure du modèle nœud mobile

La propagation dans `ns-2` est bâtie sur un modèle en 3 dimensions (x,y,z) mais ne permet pas la prise en compte des différences de propagation. La propagation est dite homogène. Les antennes sont modélisées de la sorte qu'elles rayonnent dans toutes les directions et de la même manière, mais le modèle permet d'affecter des différences de gains pour chaque antenne. Ce modèle prend aussi en compte les distances séparant les antennes. L'affaiblissement du signal lié à la distance est donc modélisé dépendamment des distances r séparant les antennes suivant les deux modèles suivants : le modèle de propagation en espace libre (*Free space model*) et le modèle de propagation utilisant deux rayons (*Two-ray ground reflection model*). Le premier modèle est utilisé si la distance r est inférieure à la distance de référence (*reference distance*) préfixée et l'atténuation est de l'ordre de $1/r^2$. Sinon, le deuxième modèle est utilisé et l'atténuation est de l'ordre de $1/r^4$.
Les liens de communication, du point de vu physique, sont modélisés de manière bi-directionnelle. L'évolution des canaux dans le temps n'est pas implémentée sur ns-2 du fait que c'est un simulateur à événements discrets.
Afin de visualiser les tracés des simulations réseaux, le simulateur `ns-2` utilise un outil d'animation qui doit être ajouté à la version de base, cet outil est le *Network AniMator* (`Nam`). Basé sur `Tcl/Tk`, il permet de visualiser tout type de réseau fixe

ou mobile permettant ainsi la visualisation en temps réel du déploiement et de la topologie des réseaux simulés.
D'autres outils peuvent être ajoutés à `ns-2`. Certains sont plus indispensables que d'autres tel que le `xgraph`, qui est un outil permettant le traçage des courbes relatives aux résultats des simulations.
Le simulateur `ns-2` propose aussi plusieurs modèles de mobilité indispensables pour simuler les réseaux mobiles. Ces modèles reproduisent les caractéristiques dynamiques de ce type de réseaux en variant les vitesses ainsi que les directions des nœuds du réseau simulé.

4.1.3 Spécificités et modifications

Pour les fins de la simulation et sur le conseil des concepteurs de l'algorithme de recherche de route multichemins[2], nous avons désactivé la recherche par expansion en anneau *(expanding ring search)* lors de la phase de recherche et de découverte de route. La recherche par expansion en anneau n'est autre qu'une technique complémentaire à la technique de recherche de route multichemins. Ceci est supposé simplifier la lecture et l'analyse des résultats des simulations.
Tout comme pour la norme 802.11, la détection des liens rompus se fait à l'aide de la réception des messages *HELLO*. Si aucun message *HELLO* n'est reçu à partir du nœud voisin pour une période de temps bien définie, le lien reliant ce nœud et son nœud voisin est considéré rompu.
Il est à noter qu'à la différence du protocole AOMDV sur lequel nous nous sommes basés pour trouver et maintenir plusieurs routes disjointes reliant un nœud source et un nœud destination, nous utilisons plusieurs routes à la fois. Cette modification est nécessaire à la garantie de l'intégrité de données.

4.1.4 Limites de la simulation

Contrairement aux travaux réalisés afin de déterminer les performances d'un protocole orienté réseau comme les protocoles de routage à titre d'exemple, nous travaillons ici sur un protocole orienté sécurité. Même si ce protocole se base sur un protocole de routage multichemins, cette composante ne représente qu'une seule partie du protocole multichemins en entier. Cette différence nous oblige à limiter nos

[2]Site web : http ://www.cs.sunysb.edu/~mahesh/aomdv/

expérimentations dans certains cas, du fait que certaines composantes du protocole ne sont pas prises en charge par les simulateurs en général et non pas spécialement ns-2. Il nous a fallu aussi trouver des moyens détournés pour récupérer les résultats dans d'autres situations.

4.1.4.1 Composantes non simulées

Au niveau des simulations, nous nous sommes retrouvés face à un degré d'abstraction ne permettant pas de prendre en compte certaines applications. L'application responsable du hachage des paquets, de leur ordonnancement et leur préparation à l'envoi dans l'ordre voulu n'est pas ou est très difficilement implantable sur le simulateur. D'autre part, nous n'avons pas réussi à implémenter la fonction de vérification responsable de la comparaison des paquets et des hachés leur correspondant, afin de valider ou d'invalider un paquets donnés. Ce type d'application devrait être implémenté au niveau de la couche application.
À cet effet, cette partie du protocole sera ignorée lors de nos simulations, les hachés des paquets seront remplacés par l'envoi de données de même taille en vue de pouvoir estimer les latences sur le réseau. La vérification de la correspondance entre les paquets et leurs hachés respectifs ne sera pas effectuée en temps réel. Elle sera remplacée par la lecture des logs, afin de déterminer si une donnée a, oui ou non, transitée par un ou plusieurs nœuds malicieux et de déterminer ainsi si les données ont été corrompues. Cette lecture ainsi que les déductions sont possibles grâce au fait que les nœuds malicieux du réseau sont déjà pré-identifiés comme tels.

4.2 Simulations et plan d'expériences

Dans cette section nous allons décrire la mise en œuvre de l'environnement des simulations à effectuer ainsi que du plan d'expérience à suivre. Nous allons aussi définir les métriques utilisées à cette fin et énoncer les différents scénarios envisagés. Ceci nous permet de définir les indices de performance que nous allons pouvoir mesurer afin de pouvoir évaluer les performances du modèle proposé.

4.2.1 Configuration de la simulation

Afin de mener à bien nos simulations, il est impératif de fixer l'environnement physique dans lequel vont se dérouler les différentes simulations à faire. Il est à noter que certaines conditions de simulation citées ci-dessous sont inspirées des conditions de simulations de Das et Marina (2001). Ce choix est motivé par la volonté de travailler dans les conditions stables et optimales du modèle de recherche et de maintien des routes multichemins.
Nous considérons donc un rectangle de dimensions 1000 mètres par 1000 mètres. Ce rectangle représente l'aire dans laquelle serons confinés les déplacements des nœuds du réseau à simuler. Nous considérons aussi 100 nœuds représentant la totalité du réseau sur lequel seront menées les évaluations de performance du modèle proposé. Ces nœuds seront initialement placés suivant une fonction aléatoire uniforme pour garantir une bonne distribution des nœuds dans le champs alloué. Cela permet d'éviter l'encombrement des nœuds sur une portion du champs.

```
# Default Script Options
# ======================================================================
set opt(chan)           Channel
set opt(prop)           Propagation/TwoRayGround
set opt(netif)          Netif/SharedMedia
set opt(mac)            Mac/802_11
set opt(ifq)        PriQueue
set opt(ll)         LL
set opt(ant)          Antenna/OmniAntenna

set opt(nn)         100         ;# number of nodes
set opt(x)          1000        ;# X dimension of the topography
set opt(y)          1000        ;# Y dimension of the topography
set opt(cp)         "./scen/traffic/udp-cbr-100-50-1-512" ;# connection pattern file
set opt(sc)         "./scen/mobility/scen-mean-1000x1000-100-0-5-1" ;# scenario file
|
|
|
set opt(ifqlen)             50          ;# max packet in ifq
set opt(seed)               0.0
set opt(stop)               1000.0              ;# simulation time
set opt(tr)         "out.tr"            ;# trace file
set opt(rp)          "multipath.tcl"            ;# routing protocol script
|
|
'
```

FIGURE 4.2 Fichier d'initialisation des paramètres de simulation run.tcl

Le déplacement des nœuds sur le champ suit aussi une fonction aléatoire four-

nie par ns-2 appelée *The random waypoint mobility model.* Ceci permet d'avoir un déplacement imprévisible des nœuds ce qui nous garanti d'innombrables situations à étudier.

Afin de garantir une mobilité continue des nœuds, nous avons choisi de fixer le temps de pause à la valeur zéro, ceci afin de pouvoir évaluer le modèle face à un réseau dynamique et constamment en mouvement. La vitesse des nœuds étant variable, et différente d'un nœud à un autre, nous faisons alors varier leur vitesse moyenne v suivant une fonction aléatoire uniforme qui choisi la vitesse du nœud dans l'intervalle $[0.9v, 1.1v]$, ceci a pour effet de faire varier continuellement le taux de mobilité du réseau simulé.

Le trafic généré sur le réseau suit, quant à lui, un patron qui se compose de plusieurs connexions qui sont établies aléatoirement entre plusieurs paires de nœuds sources et nœuds destinations. Les paquets échangés sont sous deux formats, le premier format représente les données à transmettre, le deuxième format représente les hachés des paquets. Les paquets et les hachés ont des tailles fixes qui sont respectivement de 512 octets et de 20 octets. La taille des hachés est le résultat de la fonction de hachage `SHA-1` qui pour n'importe quel longueur de données d'entrée produit un haché de 20 octets.

Les connexions commencent à différents instants choisis eux aussi aléatoirement durant les premières 100 secondes de la simulation. La durée de chaque connexion n'est pas fixe mais les connexions sont toutes maintenues du moment de leur établissement jusqu'à la fin de la simulation.

Même si la durée d'une simulation est de 1000 secondes, l'évaluation ne se fait que sur 750 secondes. La raison est que les premières 250 secondes sont utilisées comme période de démarrage et de stabilisation du réseau. Le fait d'ignorer cette première partie de la simulation nous garanti des résultats plus cohérents car les nœuds et les connexions auront eu le temps d'être mieux répartis.

Comme déjà cité ci-dessus, nous utilisons un modèle de communication sans-fil qui a un débit nominal de 2 Mb/s et une porté radio de 250 mètres. Nous avons fait le choix d'utiliser un modèle qui n'implémente pas les erreurs dues aux interférences sur les canaux. Ce choix nous permet de mieux isoler le comportement du protocole à étudier en évitant la confusion d'interprétation qui découlerait des erreurs reliées aux interférences radio. Le modèle utilisé implémente des antennes omnidirectionnelles qui permettent d'atteindre tous les nœuds se trouvant dans un rayon inférieur ou

```
# set up the antennas to be centered in the node and 1.5 meters above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to make
# it work like the 914MHz Lucent WaveLAN DSSS radio interface
Netlf/SharedMedia set CPThresh_ 10.0
Netlf/SharedMedia set CSThresh_ 1.559e-11
Netlf/SharedMedia set RXThresh_ 3.652e-10
Netlf/SharedMedia set Rb_ 2.0e6
Netlf/SharedMedia set Pt_ 0.2818
Netlf/SharedMedia set freq_ 914e+6
Netlf/SharedMedia set L_ 1.0

# the above parameters result in a nominal range of 250m
set nominal_range 250.0
set configured_range -1.0
set configured_raw_bitrate -1.0
```

FIGURE 4.3 L'initialisation des paramètres antenne dans run.tcl

égale à sa portée soit 250 mètres. De plus, ce choix n'a aucune influence sur le protocole SDMRP car les interférences radio causant des défauts de transmission sont prises en charge au niveau des couches 2 et 4 du modèle TCP/IP.

4.2.2 Métriques et indices de performances

Dans cette section, nous allons définir les métriques et les indices de performances que nous utiliserons afin de mener à bien l'évaluation du modèle proposé. Nous allons évidemment reprendre les indices de performances étudiés théoriquement dans le chapitre précèdent, tel que :

- L'évolution de la latence moyenne en fonction du nombre n de routes utilisées qui représente le délai moyen que met un groupe donné, composé d'un paquet et de ses $(n-1)$ hachés pour transiter entre un nœud sources et un nœud destination.
- L'évolution de la probabilité de succès d'une attaque en fonction du nombre de routes utilisées qui représente la probabilité de succès qu'a un attaquant possédant l'ensemble de tous les nœuds malicieux du réseau et ayant la capacité de les faire tous coopérer. Ceci représente le pire cas.

Le fait de reprendre ces deux indices de performances nous permet de comparer les résultats obtenus théoriquement aux résultats pratiques et de conforter nos conclusions.
Il est à noter que pour des raisons de simplification de calculs, les métriques utilisées dans le chapitre précedent dans la section étude théorique, se basent seulement sur le nombre de routes utilisées et ne prennent pas en compte les autres variables tel que la vitesse des nœuds, le nombre de connexions établies entre les nœuds sources et les nœuds destinations ou encore le taux de trafic généré sur chaque connexion ainsi que le nombre minimal de routes que doit trouver l'algorithme de recherche de routes.

4.2.3 Plan d'expérience

Dans le cadre de notre travail de recherche, nous essayons d'exploiter l'une des propriétés qu'offre la nature des réseaux Ad-hoc mobiles afin de la retourner contre d'éventuels attaquants. L'existence de plusieurs routes reliant un nœud source à un nœud destination est particulièrement l'avantage qu'offrent les réseaux Ad-hoc mobiles et que nous avons l'intention d'exploiter.
La variable principale sur laquelle nous devons donc nous focaliser est le nombre n de routes reliant un nœud source à un nœud destination. Cette variable est la plus importante car c'est principalement d'elle que dépend le succès et l'échec d'une éventuelle attaque sur l'intégrité des données. Elle est aussi directement reliée à la variation de la latence moyenne sur le réseau.
Dans le travail d'expérimentation qui va suivre, le nombre de routes n sera donc la variable principale. D'autres variables seront aussi prises en considération telles que :

- La vitesse moyenne des nœuds : En variant la vitesse moyenne des nœuds nous faisons varier la mobilité des nœuds. Nous touchons donc au taux de mobilité du réseau. Cette mobilité est susceptible d'engendrer plus de ruptures de liens mettant à l'épreuve le processus de recherche de routes.
- Le nombre de connexions : Nous faisons varier le nombre de connexions entre les paires de nœuds sources et destinations. Ceci est susceptible d'augmenter la charge des nœuds intermédiaires et de se répercuter sur la latence.
- Le taux d'envoi : le taux d'envoi de paquets est une autre variable susceptible, tout comme le nombre de connexions, de faire augmenter la charge du réseau.

influant ainsi sur la latence.

- La charge moyenne : la charge moyenne est le nombre moyen de bits circulant sur le réseau par seconde.
- Le temps de pause : le temps de pause est le temps pendant lequel tous les nœuds ne bougent pas et donc le réseau est temporairement statique.

Il est à noter que chaque valeur qui sera affichée dans les résultats à venir représente une moyenne de plusieurs valeurs obtenues grâce à plusieurs exécutions et ayant chacune un scénario de mobilité différent, généré d'une manière aléatoire, mais ayant le même nombre n de routes ainsi que la même vitesse moyenne des nœuds.
Le fait d'avoir recours aux exécutions multiples nous donne la garantie d'éviter de tomber dans des situations particulières. Cela nous permet donc une meilleur lecture des performances du modèle proposé.
Nous avons remarqué, lors des différentes exécutions, que les résultats obtenus se situent sensiblement dans un intervalle serré. L'écart type entre les différentes valeurs est donc assez réduit.
Il est aussi à noter que le changement des variables est opéré une à la fois, c'est-à-dire d'une série d'exécution à une autre, une seule variable est changée à la fois.
Ceci nous évite les confusions et nous permet d'isoler les variations dans le comportement du modèle proposé et pouvoir ainsi les attribuer à un changement de variable bien défini.

4.3 Simulations et analyses de résultats

Dans cette section, nous allons évaluer les performances du protocole proposé par rapport au protocole AODV et ce face à différents scénarios où l'on fait varier la mobilité, le trafic, ainsi que les nombre n de routes utilisées. Ceci dans le but de mettre en évidence les deux aspects principaux que nous avons l'intention d'étudier, à savoir :

- La probabilité de succès d'une attaque sur l'intégrité des données d'un message donné en fonction du nombre de routes utilisées ainsi que du nombre des nœuds corrompus possédées par l'attaquant.
- La variation de la latence engendrée par l'augmentation du nombre de routes utilisées.

4.3.1 Étude de la latence

Nous commençons nos simulations par l'étude de la latence sous différents scénarios que nous examinerons, par la suite, en détail :

1. Variation de la mobilité : Ce scénario consiste à étudier la latence en fonction de la mobilité des nœuds et donc de leur vitesse de déplacement
   - Réseau statique : Aucune mobilité n'est notée et la vitesse des nœuds est de 0 m/s.
   - Réseau mobile : Les nœuds sont mobiles et leur vitesse varie de 0 m/s à 30 m/s.
2. Variation du nombre de connexions : Ce scénario consiste à étudier la latence en fonction du nombre de connexions entre chaque paire de nœuds dans le réseau.
3. Variation de la charge : Ce scénario consiste à étudier la latence en fonction de la charge du réseau et donc en fonction du taux d'envoi de paquets entre les nœuds.

4.3.1.1 Variation de la mobilité

Pour commencer nous faisons varier la mobilité des nœuds. Ceci se traduit par la variation de la vitesse de déplacement des nœuds dans le réseau. Dans un premier temps, nous allons étudier la latence dans un réseau statique où la mobilité est nulle. Nous étudierons, par la suite, la variation de la latence quand les nœuds du réseau se déplaceront suivant une vitesse de 0 m/s à 30 m/s.

Réseau statique Afin de pouvoir comparer les résultats des simulations avec ceux obtenus par calculs dans le chapitre précèdent, nous commençons par une mobilité nulle. Ceci car dans les calculs effectués au chapitre précédent et, pour des raisons de simplifications, nous n'avons pas considéré la mobilité des nœuds et nous nous sommes contentés d'évaluer la variation de la latence due à l'augmentation du nombre n de routes utilisées dans un environnement statique.
À cet effet nous choisissons les valeurs de simulations suivantes :

- Vitesse max des nœuds = 0 m/s
- Nombre de connexions = 30
- Charge moyenne pour SDMRP = 130 kb/s

- Taux d'envoi de paquets/connexion pour SDMRP = n paquets/s dont 1 paquet de 512 octets et $n-1$ paquet de 20 octets chacune
- Temps de pause = 0 s

La latence calculée ne représente pas le délai bout à bout et n'inclut donc pas les délais de recherche et de maintien des routes. Ce choix est fait afin de pouvoir comparer ces résultats avec les résultats théoriques qui n'incluent pas les délais de recherche et de maintien des routes. Pour ce fait, nous fixons la vitesse max des nœuds du réseau à zéro mètres par seconde ce qui implique une mobilité nulle.

Sur la figure 4.4, chaque point représente une valeur moyenne obtenue grâce à cinq (5) exécutions. Nous remarquons que l'allure des latences moyennes trouvées expérimentalement est significativement proche de l'allure des valeurs trouvées par les calculs théoriques. La variance des valeurs obervées est de $\sim 4,2\%$ et l'écart type des échantillons est de $\sim 3,3\%$.

Toutefois, sur les simulations, nous n'avons pas pu atteindre un nombre de routes aussi grand que celui que nous avons considéré dans nos calculs théoriques (30 routes) et ce pour une raison très simple : la taille limitée du réseau simulé et l'exigence de routes complètement disjointes nécessaires au fonctionnement du protocole SDMRP (*Secure Disjoint Multipath Routing Protocol*), limitent ce nombre, nous pouvons d'ailleurs remarquer une légère inflexion sur la courbe à hauteur de $n = 8$ routes, cette inflexion est due au manque de routes disjointes en nombre suffisant et s'explique par les délais de recalcul de routes occasionné par cette difficulté.

Dans le cadre de nos simulations, il était déraisonnable de vouloir trouver plus que 10 routes disjointes ayant une longueur raisonnable et reliant n'importe quelle paire de nœuds du réseau. D'autre part, la latence moyenne reliée à un nombre de routes plus gros que 10 commencait à poser de sérieux problèmes reliés au timeout.

En effet le délais moyen s'approchait dangereusement de la valeur du timeout fixé par le protocole, ce qui augmentait fortement le risque de déstabiliser l'équilibre du protocole en l'entraînant dans une boucle sans fin de demandes de renvoi causées par le dépassement du timeout. C'est la raison pour laquelle nous nous arrêtons spécialement au chiffre 10.

Dans la figure 4.4, nous pouvons noter que la latence moyenne augmente avec l'accroissement du nombre de routes empruntées, mais que cette latence n'augmente pas linéairement, l' augmentation a tendance à diminuer avec l'accroissement du nombre de routes.

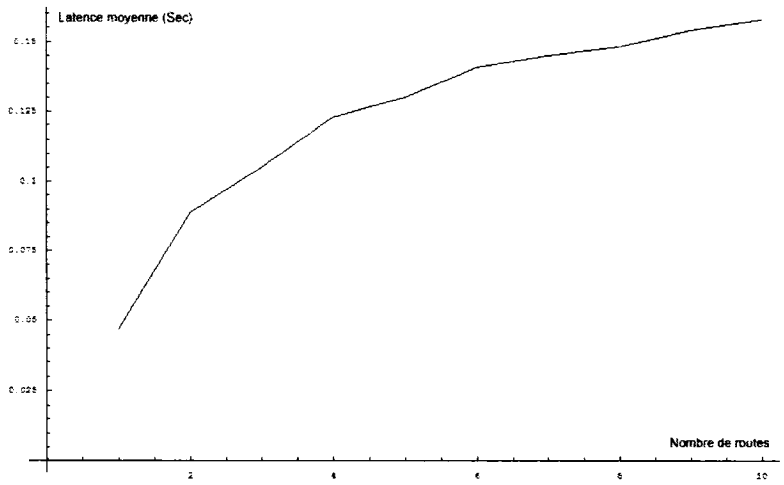


FIGURE 4.4 valeur moyenne de la latence en fonction du nombre de routes

Réseau mobile Afin de pouvoir comparer les performances du protocole SDMRP (*Secure disjoint multipath routing protocol*) avec le protocole AODV, nous sommes contraints de fixer un nombre de routes sur tout le long des simulations, car le protocole AODV utilise toujours une seule route à la fois et nous ne pouvons donc pas varier le nombre de routes sur SDMRP.

Après un certains nombre d'essais, nous avons choisi de fixer le nombre n de routes utilisées à trois routes. Le nombre N de routes trouvées dépend, quant à lui, du résultat de la recherche de routes. Il n'est donc pas fixé mais devrait être supérieur à trois. Si cette condition n'est pas vérifiée, le protocole ne pourra pas fonctionner et devra donc relancer une nouvelle recherche de route. Cette situation est très peu probable dans des réseaux denses tel que celui choisi pour cette simulation.

Ce choix nous a semblé le plus judicieux contenu des conditions de simulations. Il garantit un bon compromis entre la sécurité de l'intégrité des données et les performances du protocole SDMRP.

Les simulations se déroulent avec les valeurs suivantes :

- Vitesse max des nœuds = 30 m/s
- Nombre de connexions = 30
- Charge moyenne :
  - Pour SDMRP = 130 Kb/s
  - Pour AODV = 120 Kb/s
- Taux d'envoi de paquets/connexion :

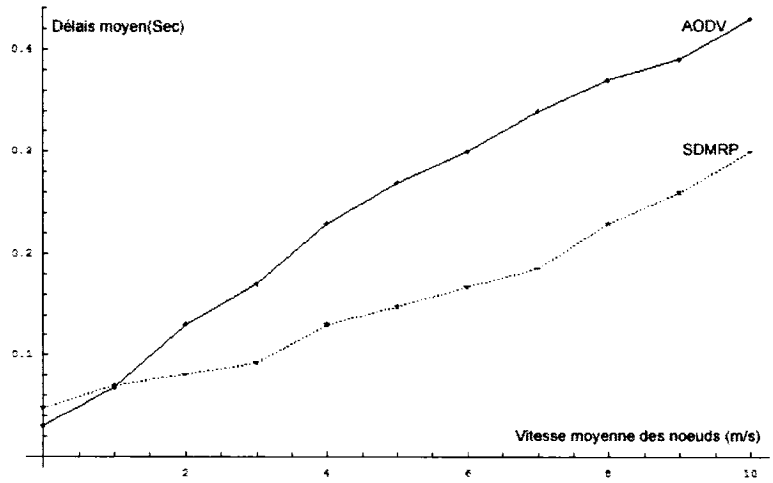


FIGURE 4.5 Latence moyenne en fonction de la vitesse moyenne des nœud

- Pour SDMRP : n paquets/s dont 1 paquet de 512 octets et $n-1$ paquet de 20 octets chaque
- Pour AODV : 1 paquets/s de taille = 512 octets

- Temps de pause = 0 s

Les vitesses des nœuds seront variées entre 0 m/s et 30 m/s qui est la vitesse max d'un nœud. Différents scénarios de mobilité seront utilisés.

Dans la figure 4.5, chaque point représente une valeur moyenne obtenue grâce à cinq (5) exécutions. La variance des valeurs obervées est de $\sim 3,4\%$ et l'écart type des échantillons est de $\sim 3\%$.

Nous pouvons voir que quand la vitesse moyenne des nœuds est basse ou nulle, le protocole AODV performe mieux que le protocole SDMRP. Ceci peut s'expliquer par le fait que SDMRP envoi plus d'informations que AODV et utilise plus de routes que ce dernier. Les deux paquets de vérification qui transitent sur deux routes différentes de la route principale sont à l'origine de cette différence.

Dès que la vitesse moyenne commence à augmenter, le protocole SDMRP commence à donner des résultats sensiblement meilleurs que ceux de AODV. Cette amélioration des performances peut être expliquée par l'augmentation de la vitesse moyenne des nœuds qui entraîne une augmentation de la mobilité du réseau. Ceci engendre plus de ruptures de routes et le délai de vie moyen des routes se trouve donc raccourci. Ce qui se traduit par plus de pertes de paquets et donc plus de demandes de retransmissions. L'algorithme de recherche de routes est alors plus souvent appelé à la rescousse, chose

qui se produit beaucoup moins souvent avec le protocole SDMRP. Cette propriété est héritée du comportement de AOMD sur lequel se base SDMRP et est due au maintien dans les tables de routage de plusieurs routes disjointes dans lesquelles l'algorithme peut puiser en cas de perte de routes. L'algorithme de recherche de route est appelé au moment d'épuisement du stock de routes disponibles.

4.3.1.2 Variation du nombre de connexions

Après avoir étudié la variation de la latence en fonctions de la mobilité, nous allons voir la variation de la latence en fonction du nombre de connexions. À cet effet, nous allons continuer à utiliser un nombre de chemins n égal à trois pour les raisons invoquées dans la section ci-dessus. Pour pouvoir isoler l'effet du nombre de connexions, la vitesse des nœuds ne sera pas variable, nous allons la fixer à 6 mètres/s.

Les autres variables garderont les mêmes valeurs que précédemment.

- Vitesse des nœuds = 6 m/s
- Nombre de connexions = de 0 à 50
- Charge moyenne :
  - Pour SDMRP = de 0 à 215 Kb/s
  - Pour AODV = de 0 à 200 Kb/s
- Taux d'envoi de paquets/connexion :
  - Pour SDMRP : n paquets/s dont 1 paquet de 512 octets et $n-1$ paquet de 20 octets chaque
  - Pour AODV : 1 paquets/s de taille = 512 octets
- Temps de pause = 0 s

Comme pour les figures précédentes, chaque point de la figure 4.6 représente la valeur moyenne de cinq exécutions. La variance des valeurs obervées est de $\sim 4,4\%$ et l'écart type des échantillons est de $\sim 3,7\%$.

Cette figure nous donne un comparatif de la variation de la latence moyenne en fonction du nombre de connexions qui varie de 0 à 50. La figure 4.6 nous montre que les performances du protocole SDMRP sont très comparables à ceux du protocole AODV tant que le nombre de connexions reste au-dessous de 15. Dès que le nombre de connexions dépasse 15, le protocole SDMRP commence à supplanter le protocole AODV. Cette différence de performances peut être expliquée par les mêmes arguments que ceux données à la section précédente c'est-à-dire le comportement de

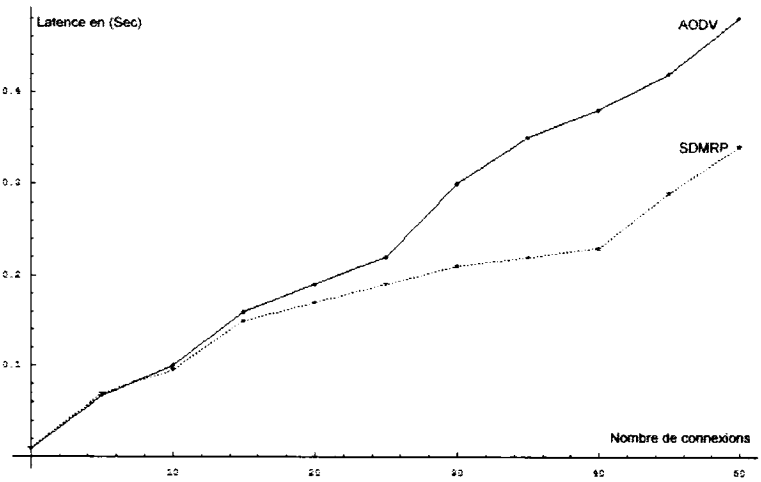


FIGURE 4.6 Latence moyenne en fonction du nombre de connexions

AOMDV.

Tout comme pour l'augmentation de la vitesse des nœuds, l'augmentation du nombre de connexions augmente le nombre de route utilisées, ce qui se traduit par plus de perte de routes. Le fait que SDMRP maintien des routes de secours lui procure un avantage assez sérieux sur AODV. Il peut plus facilement remplacer une route brisée en puisant dans sa réserve, ce qui lui permet d'éviter de relancer la recherche de route pour chaque route perdue. Ceci explique la latence plus faible de SDMRP.

4.3.1.3 Variation de la charge

Dans cette partie nous nous intéressons à l'effet de l'augmentation de la charge moyenne du réseau et son implication sur les performances des deux protocoles AODV et SDMRP. Comme pour les simulations précédentes, nous allons étudier la variation de la latence moyenne car cette dernière nous semble un très bon indicateur du comportement d'un protocole donné et résume, à elle seule, plusieurs phénomènes. Il est à noter que les délais occasionnés par le temps de calcul des hachés ainsi que le temps des comparaisons ne sont pas significatifs relativement aux délais moyens sur le réseau et n'ont donc pas d'implication sensible sur le comportement du protocole.

Pour les besoins des simulations, nous allons utiliser les variables suivantes :

- Vitesse des nœuds = 6m/s
- Nombre de connexions = 30

- Charge moyenne :
  - Pour SDMRP = de 32,5 à 194 Kb/s
  - Pour AODV = de 30 à 180 Kb/s
- Nombre n de routes pour SDMRP = 3
- Temps de pause = 0s

L'augmentation de la charge se fera grâce à l'augmentation du taux d'envoi des paquets. Les paquets garderont la même taille que précédemment soit 512 octets pour AODV. Quant à SDMRP deux tailles de paquets sont prévues : 512 octets pour le paquet principal et 20 octets pour chaque paquet de contrôle contenant le haché du paquet principal.
La variation du taux d'envoi se fera dans une plage qui va de 0.25 paquet/s à 1.5 paquet/s, ce qui se traduit pour SDMRP par 0.25 paquet/s à 1.5 paquet/s de taille 512 octets et 0.25 n paquet/s à 1.5 n paquet/s de taille 20 octets. Avec n nombre de routes = 3.

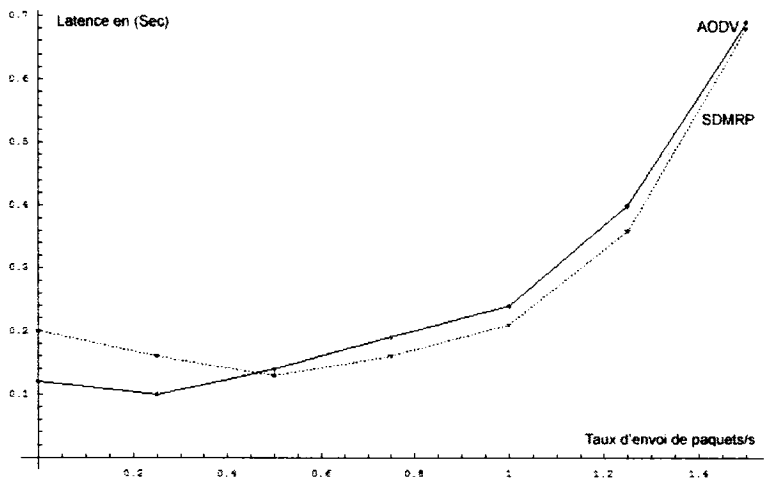


FIGURE 4.7 Latence moyenne en fonction du taux d'envoi

La figure 4.7 nous montre l'évolution de la latence moyenne sur le réseau en fonction du taux d'envoi de paquets/s. Chaque point de la figure 4.7 représente la valeur moyenne de cinq exécutions. La variance des valeurs obervées est de $\sim 4,1\%$ et l'écart type des échantillons est de $\sim 3,6\%$.
Dans cette figure, nous pouvons remarquer qu'à très faible taux d'envoi, le protocole AODV performe mieux que le protocole SDMRP. Cette différence peut être expliquée par le fait que chaque fois que le protocole AODV envoie un paquet sur

une route, le protocole SDMRP envoie trois paquets sur trois routes. Dans les simulations précédentes le protocole SDMRP se rattrapait et dépassait AODV grâce à l'utilisation des routes de réserves qu'il maintient en permanence.
Quand le taux d'envoi des paquets est bas, cet avantage n'a plus de raison d'être. En effet, plus ce taux d'envoi est bas, plus le temps séparant deux envois est grand. Si ce temps devient assez grand, la plupart des routes maintenues en réserve par SDMRP seront périmées à cause de la mobilité du réseau. Même si la mobilité est faible (6m/s), le grand délai séparant les envois donne le temps aux nœuds de parcourir d'assez grandes distances, les menant hors de la porté antenne de leurs nœuds voisins. SDMRP perd donc son avantage habituel sur AODV et ne garde que son inconvénient qui est l'envoi de plus de paquets sur plus de routes.
Dès que le taux d'envoi de paquets augmente, SDMRP commence à mieux performer que AODV. Il est à noter qu'une grande augmentation du taux d'envoi des paquets s'accompagne d'une dégradation des performances des deux protocoles, qui s'avèrent avoir des performances comparables dans ce cas de figure.

4.3.2 Étude de la sécurité

Comme nous l'avons fait pour l'étude de la latence moyenne sur le réseau, nous allons ici aussi faire la comparaison entre les résultats obtenus par calcul dans le chapitre précédent et ceux obtenus grâce aux simulations. Toutefois, le simulateur ne nous permet pas de mener ces expériences d'une façon directe. En effet, il n'y a aucun moyen de définir un nœud du réseau comme étant un nœud malicieux. D'autres part, le simulateur ne permet pas de changer le contenu des données transitant par un nœud donné. L'autre chose à prendre en considération est le fait que l'algorithme de vérification de la concordance des paquets avec leurs hachés, ainsi que celui responsable de la création des hachés ne sont pas implémentés. Il n'y a d'ailleurs aucun moyen de spécifier le degré de collaboration des nœuds. Pour ce faire, nous fixons alors les nœuds que nous allons considérer comme malicieux ainsi que la capacité des nœuds à collaborer. Suite à quoi, nous faisons rouler les simulations. À la fin du déroulement de la simulation, nous pouvons regarder et analyser les traces d'exécutions pour déterminer si une attaque a eu lieu, et si cette dernière a réussi ou a échoué. Pour qu'une attaque soit considérée comme réussie, la présence d'au moins un nœud malicieux sur chacune des routes ayant servi à une même connexion

est nécessaire. Il est à noter que sans collaboration entre les nœuds aucune attaque ne pourra réussir. La condition de collaboration de tous les nœuds participant à une attaque est donc nécessaire pour la réussite de cette dernière.
Nous allons, dans ce qui suit, évaluer les performances de notre protocole et évaluer le risque de réussite d'une attaque suivant les scénarios suivant :

1. Évaluation du risque en fonction du nombre de routes n.
2. Évaluation du risque en fonction du taux de corruption des nœuds.

4.3.2.1 Évaluation du risque en fonction du nombre de routes n

Dans cette section, nous allons étudier l'évolution de la probabilité de succès d'une attaque en fonction du nombre n des routes utilisées par le protocole SDMRP. Afin de réaliser les simulations nous allons utiliser les variables suivantes :

- Vitesse des nœuds = 6 m/s
- Nombre de connexions = 30
- Nombre de routes n : à faire évoluer de 1 à 10
- Taux d'envoi de paquets/connexion :
  - Pour SDMRP : n paquets/s dont 1 paquet de 512 octets et $n-1$ paquet de 20 octets chaque
  - Pour AODV : 1 paquets/s de taille = 512 octets
- Temps de pause = 0 s

Nous allons étudier le pire cas, c'est-à-dire que nous allons considérer les hypothèses les plus défavorables. Nous considérons donc une collaboration entre tous les nœuds malicieux du réseau. Dans le cadre de ces simulations, nous allons utiliser des durées de 5000 secondes par simulation et effectuer 20 exécutions. Ceci devrait nous permettre le recueil d'informations et sur une période assez longue, ce qui est meilleur pour l'analyse de la probabilité de succès d'une attaque.
Le taux des nœuds malicieux est maintenu à 0.2, ce qui veut dire qu'un nœud sur cinq du réseau est malicieux. Les nœuds malicieux sont choisis aléatoirement au début des simulations. Un nœud, qui au début de la simulation est considéré comme malicieux, le reste jusqu'à la fin de celle-ci, de même pour un nœud non malicieux. Au cours des simulations il n'y a pas d'ajout ou de diminution de nœuds.

Dans la figure 4.8, nous pouvons observer la détérioration des chances de succès de l'attaquant en fonction de l'augmentation du nombre n de routes utilisées. Cette

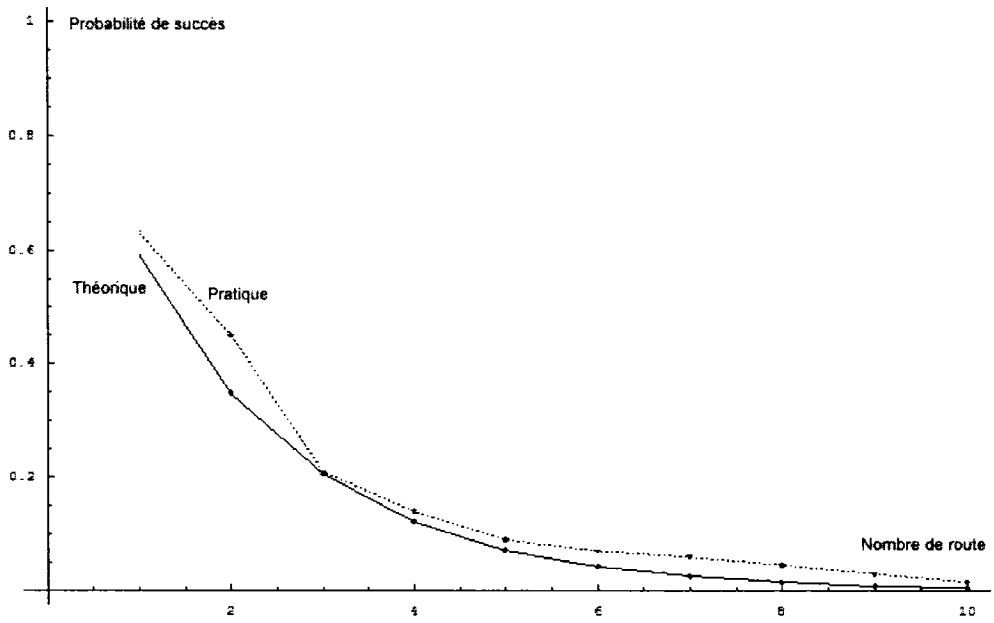


FIGURE 4.8 Évaluation de la probabilité de succès d'une attaque en fonction de n

courbe obtenue grâce aux simulations s'approche beaucoup de celle obtenue théoriquement par la formule $(1-(1-p)^l)^n$, où l représente la longueur des routes, p la probabilité qu'un nœud soit malicieux et n le nombre de routes utilisées, ce qui conforte nos calculs.

Rendu à $n = 10$, les chances de succès d'une attaque sont pratiquement nulles. L'évolution de la probabilité de succès est inversement exponentielle au nombre n des routes.
Avec un taux de nœud malicieux = 0.2, l'utilisation d'une seule route, comme pour le cas du protocole AODV, donne à l'attaquant une chance de succès approximative de 0.6. En utilisant le protocole SDMRP avec un nombre de routes $n = 3$, cette chance tombe à 0.2.
Nous remarquons qu'au-delà d'un nombre de route n supérieur à dix les chances de succès d'une attaque sont presque nulles. Il n'y a donc pas de gain à choisir un nombre de route plus grand. D'autre part pour un nombre de routes inférieur à dix, le protocole n'a pas besoin de recalculer les routes aussi souvent que quand n est grand, la qualité de service n'est donc pas pénalisée.

4.3.2.2 Évaluation du risque en fonction du taux de corruption des nœuds

Pour pouvoir comparer le comportement des protocoles AODV et SDMRP faces aux risques d'attaques sur l'intégrité des données, nous devons faire varier le nombre de nœud malicieux sur le réseau, c'est-à-dire le taux de corruption du réseau. Pour cela nous allons comparer AODV avec trois variantes de SDMRP. La première sera avec un nombre de routes $n = 3$, la deuxième sera avec un nombre de routes $n = 6$, la troisième avec $n = 9$.

Nous ferons varier le taux de compromission du réseau de 0.0 à 0.5 où un taux = 0 voudrai dire qu'aucun nœud du réseau n'est malicieux et un taux = 1 que tous les nœuds du réseau sont malicieux. Pour les besoins des simulations, nous allons considérer le pire cas qui correspond à l'hypothèse où tous les nœuds du réseau coopèrent à tout moment et sans conditions.

Les variables suivantes seront utilisées :

- Vitesse des nœuds = 6 m/s
- Nombre de connections = 30
- Nombre de routes pour SDMRP n= 3, $n = 6$ et $n = 9$
- Taux d'envoi de paquets/connexion :
  - Pour SDMRP : n paquets/s dont 1 paquet de 512 octets et $n - 1$ paquet de 20 octets chaque
  - Pour AODV : 1 paquets/s de taille = 512 octets
- Temps de pause = 0 s

Dans la figure 4.9, nous pouvons observer et comparer les performances du protocole SDMRP et du protocole AODV où chaque point de la figure représente la moyenne de 20 exécutions sur une durée de 5000 secondes. Nous pouvons remarquer qu'avec seulement 3 routes SDMRP surpasse AODV et procure une sécurité accrue par rapport à ce dernier.

La performance de SDMRP est très visible dans les cas les plus réalistes où l'attaquant possède moins que le cinquième du réseau. À titre d'exemple, pour une attaque où 20% des nœuds du réseau collaborent, la probabilité de succès contre AODV est de $0,6$ alors qu'elle est respectivement de $0,2$, $0,05$ et inférieur à $0,01$ pour SDMRP avec 3, 6 et 9 routes. Il est à noter qu'en faisant l'hypothèse de la collaboration inconditionnelle de tous les nœuds malicieux du réseaux, nous pénalisons

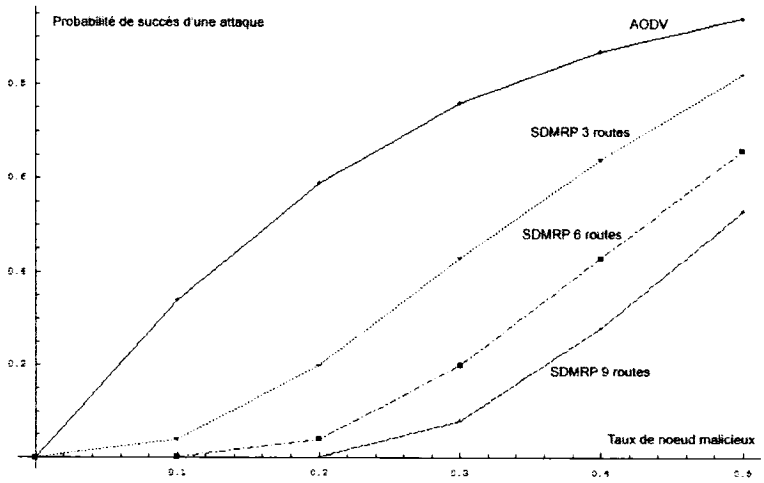


FIGURE 4.9 Probabilité de succés d'attaque en fonction du taux de nœuds malicieux

seulement le protocole SDMRP, car même si les nœuds malicieux ne collaboraient pas les performances de AODV resteraientt les mêmes, ce qui n'est pas le cas de SDMRP.

CHAPITRE 5

Conclusions

Dans ce travail de recherche, nous avons essayé de traiter les problèmes entourant la sécurité des réseaux Ad-hoc mobiles et plus précisément les problèmes reliés à la garantie de l'intégrité des données.
Pour cela, nous avons proposé une stratégie qui se base sur l'idée d'exploiter l'une des propriétés des réseaux Ad-hoc mobiles, à savoir l'existence éventuelle de plusieurs routes reliant un nœud source à un nœud destination. Afin de mettre en œuvre cette idée, nous avons développé un protocole de routage basé sur l'utilisation de l'envoi multichemins disjoints. L'objectif principal de ce protocole que nous avons nommé SDMRP *(secure disjoint multipath routing protocol)* est de garantir une amélioration de l'intégrité des données en rendant plus difficile, voir impossible, la tâche de l'attaquant.

5.1 Synthèse des travaux

Les travaux réalisés dans ce mémoire nous ont permis d'assurer une meilleure garantie de l'intégrité des données transitant sur les réseaux Ad-Hoc mobile et d'améliorer, de ce pas, la sécurité dans ces réseaux. En effet, le protocole SDMRP proposé permet de diminuer exponentiellement par rapport au nombre de routes n la probabilité de succès d'une attaque coordonnée. Cette probabilité passe donc de $1-(1-p)^l$ à $(1-(1-p)^l)^n$ où l est la longueur du chemin reliant un nœud source et un nœud destination, n le nombre de routes utilisées et p la probabilité qu'un nœud du réseau soit malicieux. Ces résultats ont été vérifiés autant théoriquement qu'expérimentalement. De plus, SDMRP éradique complètement les attaques non coordonnées et sans collaboration sur l'intégrité des données, faisant passer leur probabilité de succès de $1-(1-p)^l$ à 0.
Nous remarquons aussi que le protocole SDMRP ne pénalise pas la qualité de service sur les RAHM. En effet, nous avons démontré dans notre étude théorique que la la-

tence suit une progression lente en fonction du nombre n de routes pour s'approcher à $n > 10$ d'une asymptote.
Les résultats expérimentaux corroborent, jusqu'à un certain point, les résultats théoriques mais nous avons noté que cette latence connaît un point d'inflexion à partir d'un nombre de route $n > 8$. Nous pensons que cette différence entre la théorie et la pratique est tout simplement due à la difficulté qu'à SDMRP à trouver un nombre n de routes supérieur à 10, ce qui pourrait donc l'entrainer dans une boucle de recherche et de recalcul de routes. Ce phénomène s'explique par l'existence d'un nombre fini de routes entre une source et une destination dans un réseau Ad-Hoc donné. Nous pouvons donc conclure que SDMRP présente une latence acceptable et imperceptible pour une transmission de données lorsque le nombre de routes utilisé est $3 \leq n \leq 10$.
Nous remarquons aussi que la latence de SDMRP est meilleure que AODV mais supérieure à celle de AOMDV. Ce résultat est prévisible vu que SDMRP est essentiellement basé sur AOMDV, à la différence qu'il utilise plus qu'une route à la fois. La latence est donc légèrement augmentée.
Nous avons aussi noté une diminution de la gigue lors de l'utilisation du protocole SDMRP. Ce résultat s'explique par le fait que les données transitent par plusieurs routes à la fois et que la validation de leur réception devra attendre l'arrivée de toutes les parties. SDMRP attend donc l'arrivée des données sur le plus lent des chemins avant de les accepter ce qui diminue les écarts entre les temps d'arrivée des informations.
Le protocole SDMRP présente cependant une certaine faiblesse face aux attaques par déni de service (DOS). Nous avons malheureusement constaté que sous sa forme actuelle, notre protocole est plus vulnérable à ce type d'attaque que les autres protocoles dans les réseaux Ad-Hoc. En effet, la probabilité de succès d'une attaque par déni de service passe de $1 - (1 - p)^l$ pour un protocole acheminant les données sur une seule route à $1 - (1 - p)^{ln}$ pour SDMRP.

5.2 Limitations des travaux

Pour valider nos résultats, nous avons choisis de simuler le comportement du protocole SDMRP et notre choix s'est porté sur le simulateur ns-2. Quoi qu'il soit très adapté aux simulations sur les réseaux Ad-hoc mobiles, le simulateur ns-2 possède

certaines limites. Ces limites ne lui sont pas propres et se retrouvent sur tous les autres simulateurs.

Lors de l'implémentation du protocole SDMRP sur ns-2, certaines composantes du protocole n'ont pas pu être ou étaient très difficiles à implanter, tel que la fonction de vérification de la correspondance entre les différent hachés avec le paquet dont ils sont issus, ou encore la fonction de hachage qui permet de créer les hachés à envoyer qui n'a non plus pas pu être implanté. D'autre part, nous n'avons pas pu ajouter au simulateur la capacité de spécifier un nœud comme étant un nœud malicieux. L'affectation des données transitant par un nœud donné dans le but de changer leur contenu n'a pas, non plus, été implantée.

Ces choix ont, d'une certaine manière, eu un léger impact sur les résultats obtenus surtout de point de vue de l'estimation de la latence. Nous pouvons cependant affirmer que cet impact est très peu significatif vu que les opérations de hachage et de vérification des hachés ne sont aucunement gourmandes en temps et peuvent être réalisées dans un délai presque insignifiant par rapport à la latence normale rencontrée dans le réseau.

Nous avons toutefois pu mener à bien les simulations par l'utilisation de méthodes indirectes pouvant simuler le comportement escompté et qui ont consisté à fixer préalablement les nœuds malicieux du réseau et de vérifier, par la suite, si les routes choisies par SDMRP sont passées par ces derniers. Si c'est le cas, nous affirmons que les données qui ont transité par ces nœuds ont été corrompus. Si nous trouvons qu'il existe un nœud malicieux sur chacune des routes, nous pouvons alors conclure qu'une attaque est réussie.

Durant les simulations, nous avons pu déceler un certain nombre de faiblesses reliées au fonctionnement du protocole SDMRP. Pour fonctionner correctement, SDMRP nécessite un réseau ayant une certaine densité. En effet, si le réseau est peu dense, l'algorithme de recherche et de maintien des routes éprouve de sérieuses difficultés à trouver et à maintenir un nombre minimum de routes disjointes adéquat, permettant un bon fonctionnement du protocole.

Si le réseau est peu dense, les performances de protocole en ce qui concerne la qualité de service chutent dangereusement. Cette chute est due à la multiplication des échecs que subis le protocole lorsqu'il tente de trouver et de maintenir le nombre minimal de routes exigé. La répétition incessante de demandes de recherche et de découverte de routes pour renouveler celles expirées participe aussi à cette dégradation. Le fait

que le réseau soit de faible densité offre très peu de routes disjointes. Les tables de routage contiennent alors, à peine le nombre de routes minimales requises. Dès qu'une route est perdue, l'algorithme se voit obligé de lancer une nouvelle recherche de route vu qu'il ne possède plus de réserve et perd alors son avantage par rapport à AODV.
Il est aussi très important de signaler que sous cette forme, SDMRP lutte efficacement contre les attaques sur l'intégrité des données, mais ouvre par contre une autre brèche de sécurité. Il s'agit des attaques par déni de service. Un attaquant qui n'arrive pas à modifier le contenu des données sans être découvert, oblige une retransmission complète du paquet ainsi que de ses hachés. Si le chemin sur lequel se trouve l'attaquant n'est pas écarté, le protocole entre alors dans une boucle de retransmissions sans fins. L'attaquant aura alors indirectement et sans le vouloir réussi une attaque par déni de service.
Dans le cadre de ce travail de recherche nous avons émis l'hypothèse qu'il n'y aura pas d'attaque sur l'algorithme de découverte de route. Cette hypothèse n'est pas toujours vérifiée et donc pas très réaliste.

5.3 Travaux futurs

En se basant sur ces limitations, nous pouvons entrevoir les perspectives de travaux futurs. Les simulateurs n'étant généralement pas très adaptés à l'évaluation de la sécurité, nous nous proposons de compléter l'implémentation des parties non implantées dans le cadre de ce travail. Il serait alors pertinent, quoi que pas facile à réaliser, de tester, même à petite échelle, le comportement du protocole sur un vrai réseau.
Pour améliorer les performances et le comportement du protocole SDMRP face aux attaques par déni de service, nous entrevoyons de lui amener deux modifications mineures :

- Faire une modification dans le seuil de tolérance des erreurs dans le protocole en permettant d'ignorer un seul haché différent des autres ce qui permet d'éviter une retransmission systématique du paquet et de tous ses hachés. Cette modification n'a pas un grand impact sur les performances du protocole en terme de la garantie de l'intégrité des données. Cet impact est facilement calculable. Lorsqu'on utilise cette variante, le protocole utilisant un nombre de routes n

aura exactement les mêmes performance que l'ancienne version utilisant un nombre $n-1$ de routes. Le nombre de retransmissions dues aux erreurs sur le réseau, au même titre que celles dues aux attaques, sera vu à la baisse. Ce qui a un impact positif sur la qualité de service.

- Nous prévoyons aussi d'installer l'équivalant d'une liste taboue. Cette liste pourra être utilisée afin de marquer les routes ayant un comportement inadéquat en vu de ne plus les utiliser pour une période définie. De ce fait, il devient possible d'éviter de repasser par les nœuds malicieux se trouvant sur une route déjà utilisée. Ceci risque d'améliorer les performances du protocole en ce qui concerne la sécurité mais aussi en ce qui concerne la qualité de service. Avec cette modification, nous soupçonnons la possibilité que le protocole puisse acquérir une certaine protection contre les attaques par trous noirs et trous gris. Il serait alors pertinent de prévoir des simulations en ce sens.
- Utiliser le protocole SDMRP pour le partage de clés de chiffrement telles que celles de type Diffie-Hellman pour garantir leurs intégrité. Cette clé pourra servir ensuite au chiffrement des échanges de données assurant ainsi leur intégrité et confidentialité.

En conclusion, ce travail constitue une solution valable au problème de la garantie de l'intégrité des données dans les réseaux Ad-Hoc mobiles. Elle est facile à déployer et peu contraignante pour la qualité de service dans le cas de la transmission de données. Elle trouve son application dans les réseaux de moyenne à haute densité où l'existence de plusieurs routes entre un nœud source et un nœud destination est garantie.

Références

ADJIH, C., LAOUITI, A., MINET, P., MUHLETHALER, P., QAYYUM, A. et L.VIENNOT (2003). Optimized Link State Routing Protocol (OLSR). T. Clausen et P. Jacquet, éditeurs, *RFC 3626*, IETF.

BOUAM, S. et OTHMAN, J. B. (2003). Data security in ad hoc networks using multipath routing. *Proc. IEEE Personal, Indoor and Mobile Radio Communications (PIMRC)*. 1331–1335.

BUCHEGGER, S. et BOUDEC, J.-Y. L. (2002a). Nodes Bearing Grudges : Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks. *Proc. IEEE Euromicro Workshop on Parallel, Distributed and Network-based Processing*. 403–410.

BUCHEGGER, S. et BOUDEC, J.-Y. L. (2002b). Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes : Fairness In Dynamic Ad-hoc NeTworks). *Proc. ACM Symposium on Mobile Adhoc Networking and Computing (MOBIHOC)*. 226–236.

BUTTYÁN, L. et HUBAUX, J. (2001). Nuglets : a virtual currency to stimulate cooperation in self-organized ad hoc networks. Rapport technique DSC/2001, Swiss Federal Institute of Technology.

CAPKUN, S., HUBAUX, J.-P. et BUTTYÁN, L. (2003). Mobility helps security in ad hoc networks. *Proc. ACM Symposium on Mobile Adhoc Networking and Computing (MOBIHOC)*. 46–56.

CORSON, S. et MACKER, J. (1999). Mobile ad hoc networking (MANET) : Routing protocol performance issues and evaluation considerations. *RFC 2501*, IETF.

DAS, S. R. et MARINA, M. K. (2001). On-demand multi path distance vector routing in ad hoc networks. *Proc. IEEE International Conference on Network Protocols (ICNP)*. 969–988.

GOLLE, P., GREENE, D. et STADDON, J. (2004). Detecting and correcting malicious data in VANETs. *Proc. ACM International Workshop on Vehicular Ad Hoc Networks (VANET)*. 29–37.

HAAS, Z. J., PEARLMAN, M. R. et SAMAR, P. (2002). The zone routing protocol (ZRP) for ad hoc networks. Internet-draft, IETF MANET Working Group.

HU, Y.-C. et PERRIG, A. (2004). A survey of secure wireless ad hoc routing. *IEEE Security and Privacy*, 2, 28–39.

HU, Y.-C., PERRIG, A. et JOHNSON, D. B. (2002). Ariadne : A Secure On-Demand Routing Protocol for Ad Hoc Networks. *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*. 12–23.

HU, Y.-C., PERRIG, A. et JOHNSON, D. B. (2003a). Packet leashes : A defense against wormhole attacks in wireless networks. *Proc. Annual Joint Conference of IEEE Computer Society (INFOCOM)*. 1976–1986.

HU, Y.-C., PERRIG, A. et JOHNSON, D. B. (2003b). Rushing attacks and defense in wireless ad hoc network routing protocols. *Proc. ACM Workshop on Wireless Security*. 30–40.

HUBAUX, J.-P., BUTTYÁN, L. et CAPKUN, S. (2001). The quest for security in mobile ad hoc networks. *Proc. ACM Symposium on Mobile Adhoc Networking and Computing (MOBIHOC)*. 146–155.

JOHNSON, D. B. et MALTZ, D. A. (1996). Dynamic source routing in ad hoc wireless networks. Imielinski et Korth, éditeurs, *Mobile Computing*, Kluwer Academic Publishers, vol. 353.

LEE, S. et GERLA, M. (2001). Split multipath routing with maximally disjoint paths in ad hoc networks. *Proc. IEEE International Conference on Communications (ICC)*. vol. 10, 3201–3205.

LI, X. et CUTHBERT, L. (2004). On-demand node-disjoint multipath routing in wireless ad hoc network. *Proc. IEEE Annual International Conference on Local Computer Networks (LCN)*. 419–420.

MICHIARDI, P. et MOLVA, R. (2002). CORE : a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. B. Jerman-Blazic et T. Klobucar, éditeurs, *Advanced Communications and Multimedia Security*, Kluwer Academic Publishers.

MICHIARDI, P. et MOLVA, R. (2006). Ad hoc networks security. H. Bidgoli, éditeur, *Handbook of Information Security*, Wiley & Sons.

NEWSOME, J., SHI, E., SONG, D. et PERRIG, A. (2004). The sybil attack in sensor networks : analysis & defenses. *Proc. International Symposium on Information Processing in Sensor Networks (ISPN)*. 259–268.

PERKINS, C. (1997). Ad-hoc on-demand distance vector routing. *Proc. AFCEA Military Communications Conference (MILCOM)*.

PERKINS, C. et BHAGWAT, P. (1994). Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. *Proc. ACM Conference on Communications Architectures, Protocols and Applications (SIGCOMM)*. 234–244.

PERRIG, A., CANETTI, R., TYGAR, D. et SONG, D. (2002). The TESLA broadcast authentication protocol. *Cryptobytes*, 5, 2–13.

STAJANO, F. et ANDERSON, R. (1999). The resurrecting duckling : Security issues for ad-hoc wireless networks. *Proc. ACM International Workshop on Security Protocols*. 172–194.

STALLINGS, W. (2002). *Cryptography and Network Security : Principles and Practice*. Pearson Education.

ZHOU, L. et HAAS, Z. J. (1999). Securing ad hoc networks. *IEEE Network*, 13, 24–30.